



**Question Bank for Multiple Choice Questions**

<b>Program: Diploma in Computer Engineering</b>	<b>Program Code:- CO</b>
<b>Scheme:-I</b>	<b>Semester:- II</b>
<b>Course:- Programming in C</b>	<b>Course Code:-22226</b>

Unit-1 Program Logic Development

1.1 Fundamentals of algorithms: Notion of an algorithm, pseudo-code conventions like assignment Statements and basic control structures.

1.2 Algorithmic problems: Develop fundamental algorithms to solve simple problems such as:

- (i) Solve simple arithmetic expression (ii) find the greatest of three numbers (iii) Determine whether a given number is even or odd (iv) Determine whether a given number is prime.

1.3 Flowchart: Flowchart Symbols of flowchart, Guidelines for preparing Flowchart.

1. Who is the father of C language?

- a) Steve Jobs
- b) James Gosling
- c) Dennis Ritchie**
- d) Rasmus Lerdorf

Answer: c

2. Which of the following is not a valid C variable name?

- a) int number;
- b) float rate;
- c) int variable\_count;
- d) int \$main;**

Answer: d

3. All keywords in C are in \_\_\_\_\_

- a) LowerCase letters**
- b) UpperCase letters
- c) CamelCase letters
- d) None of the mentioned

Answer: a.

4. Which of the following is true for variable names in C?

- a) They can contain alphanumeric characters as well as special characters
- b) It is not an error to declare a variable to be one of the keywords(like goto, static)
- c) Variable names cannot start with a digit
- d) Variable can be of any length

Answer: c

5. Which is valid C expression?

- a) `int my_num = 100,000;`
- b) `int my_num = 100000;`
- c) `int my num = 1000;`
- d) `int $my_num = 10000;`

Answer: b

6. Which of the following cannot be a variable name in C?

- a) `volatile`
- b) `true`
- c) `friend`
- d) `export`

Answer: a

7. What is short int in C programming?

- a) The basic data type of C
- b) Qualifier
- c) Short is the qualifier and int is the basic data type
- d) All of the mentioned

Answer: c

8. Which of the following declaration is not supported by C language?

- a) `String str;`
- b) `char *str;`
- c) `float str = 3e2;`
- d) `Both String str; & float str = 3e2;`

Answer: a

9. Which keyword is used to prevent any changes in the variable within a C program?

- a) `immutable`
- b) `mutable`
- c) `const`
- d) `volatile`

Answer: c

10. What is the result of logical or relational expression in C?

- a) True or False
- b) 0 or 1
- c) 0 if an expression is false and any positive number if an expression is true
- d) None of the mentioned

Answer: b

11. Which of the following typecasting is accepted by C language?

- a) Widening conversions
- b) Narrowing conversions
- c) Widening & Narrowing conversions
- d) None of the mentioned

Answer: c

12. Where in C the order of precedence of operators do not exist?

- a) Within conditional statements, if, else
- b) Within while, do-while
- c) Within a macro definition
- d) None of the mentioned

Answer: d

13. Which of the following is NOT possible with any 2 operators in C?

- a) Different precedence, same associativity
- b) Different precedence, different associativity
- c) Same precedence, different associativity
- d) All of the mentioned

Answer: c

14. What is an example of iteration in C?

- a) for
- b) while
- c) do-while
- d) all of the mentioned

Answer: d

15. Functions can return enumeration constants in C?

- a) true
- b) false
- c) depends on the compiler
- d) depends on the standard

Answer: a

16. Functions in C Language are always \_\_\_\_\_

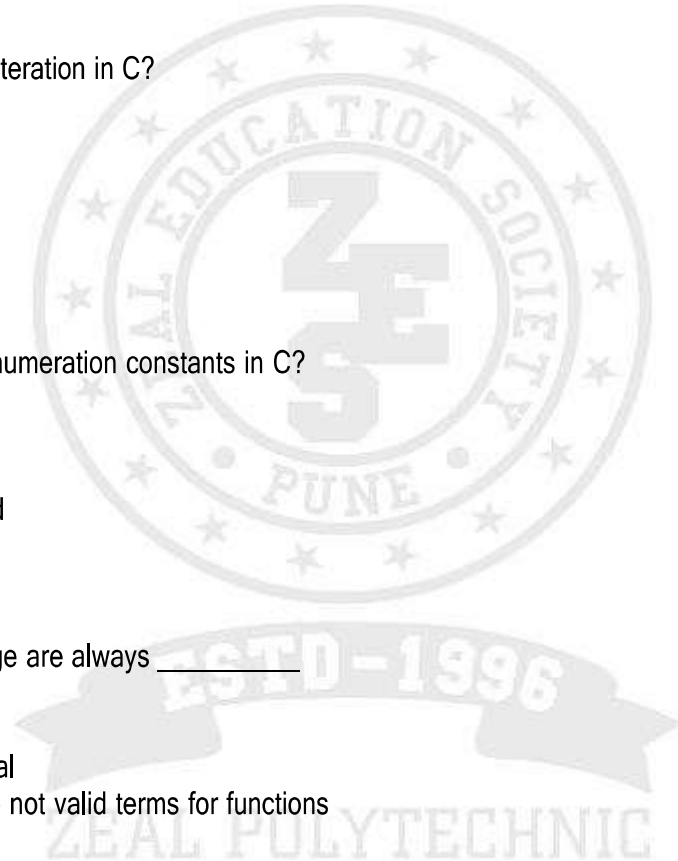
- a) Internal
- b) External
- c) Both Internal and External
- d) External and Internal are not valid terms for functions

Answer: b

17. Which of following is not accepted in C?

- a) `static a = 10; //static as`
- b) `static int func (int); //parameter as static`
- c) `static static int a; //a static variable prefixed with static`
- d) all of the mentioned

Answer: c



18. Property which allows to produce different executable for different platforms in C is called?

- a) File inclusion
- b) Selective inclusion
- c) Conditional compilation
- d) Recursive macros

Answer: c

19. What is #include <stdio.h>?

- a) Preprocessor directive
- b) Inclusion directive
- c) File inclusion directive
- d) None of the mentioned

Answer: a

20. C preprocessors can have compiler specific features.

- a) True
- b) False
- c) Depends on the standard
- d) Depends on the platform

Answer: a

21. Which of the following are C preprocessors?

- a) #ifdef
- b) #define
- c) #endif
- d) all of the mentioned

Answer: d

22. The C-preprocessors are specified with \_\_\_\_\_ symbol.

- a) #
- b) \$
- c) " "
- d) &

Answer: a

23. How is search done in #include and #include "somelibrary.h" according to C standard?

- a) When former is used, current directory is searched and when latter is used, standard directory is searched
- b) When former is used, standard directory is searched and when latter is used, current directory is searched
- c) When former is used, search is done in implementation defined manner and when latter is used, current directory is searched
- d) For both, search for 'somelibrary' is done in implementation-defined places

Answer: d

24. How many number of pointer (\*) does C have against a pointer variable declaration?
- a) 7
  - b) 127
  - c) 255
  - d) No limits

Answer: d

25. Which of the following is not possible statically in C language?
- a) Jagged Array
  - b) Rectangular Array
  - c) Cuboidal Array
  - d) Multidimensional Array

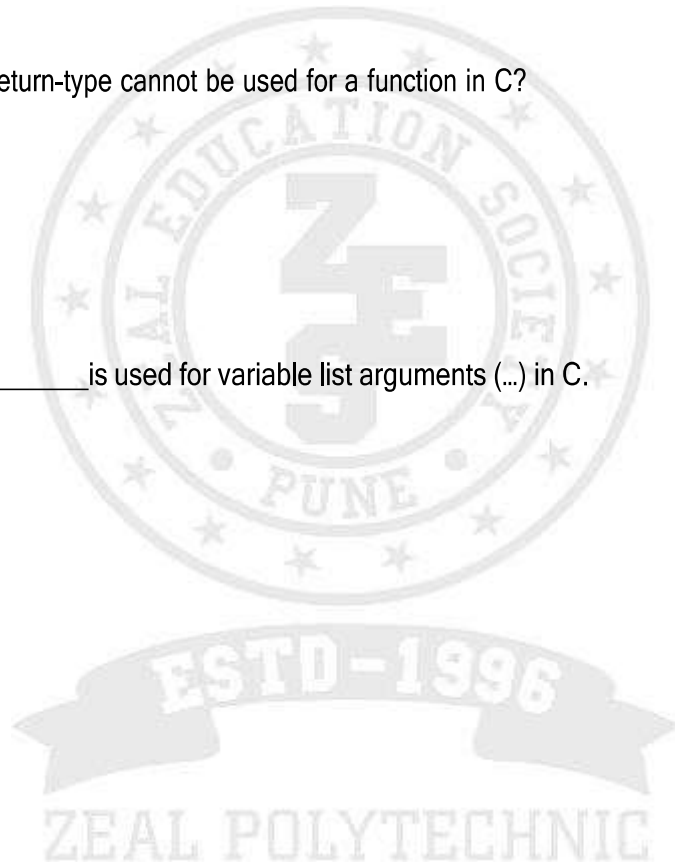
Answer: a

26. Which of the following return-type cannot be used for a function in C?
- a) char \*
  - b) struct
  - c) void
  - d) none of the mentioned

Answer: d

27. The standard header \_\_\_\_\_ is used for variable list arguments (...) in C.
- a) <stdio.h >
  - b) <stdlib.h>
  - c) <math.h>
  - d) <stdarg.h>

Answer: d



Unit-2 Basics of C programming

- 2.1 Introduction to C: History of 'C', General Structure of a 'C' program: Header files. main function.
- 2.2 Data Concept: Character set, tokens, keywords, Identifiers, Variables, Constant, data types. C operators, Arithmetic operators, Arithmetic expression, declaring variables, and data type conversion.
- 2.3 Basic input Output: input and Output statements, using printf() and scanf(). Character input/output formatting, use of comments.

28. When a C program is started, O.S environment is responsible for opening file and providing pointer for that file?

- a) Standard input
- b) Standard output
- c) Standard error
- d) All of the mentioned

Answer: d

29. In C language, FILE is of which data type?

- a) int
- b) char \*
- c) struct
- d) None of the mentioned

Answer: c

30. What is the sizeof(char) in a 32-bit C compiler?

- a) 1 bit
- b) 2 bits
- c) 1 Byte
- d) 2 Bytes

Answer: c

31. Which of the following is not an operator in C?

- a) ,
- b) sizeof()
- c) ~
- d) None of the mentioned

Answer: d

32. scanf() is a predefined function in \_\_\_\_\_ header file.

- a) stdlib. h
- b) ctype. h
- c) stdio. h
- d) stdarg. h

Answer: c

33. What is meant by 'a' in the following C operation?

```
fp = fopen("Random.txt", "a");
```

- a) Attach
- b) Append
- c) Apprehend
- d) Add

Answer: b

34. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int y = 10000;
5.     int y = 34;
6.     printf("Hello World! %d\n", y);
7.     return 0;8.
}
```

- a) Compile time error
- b) Hello World! 34
- c) Hello World! 1000
- d) Hello World! followed by a junk value

Answer: a

35. What will happen if the following C code is executed?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int main = 3;
5.     printf("%d", main);
6.     return 0;7.
}
```

- a) It will cause a compile-time error
- b) It will cause a run-time error
- c) It will run without any error and prints 3
- d) It will experience infinite looping

Answer: c

36. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     signed char chr;
5.     chr = 128;
6.     printf("%d\n", chr);
7.     return 0;
8. }
```

- a) 128
- b) -128
- c) Depends on the compiler
- d) None of the mentioned

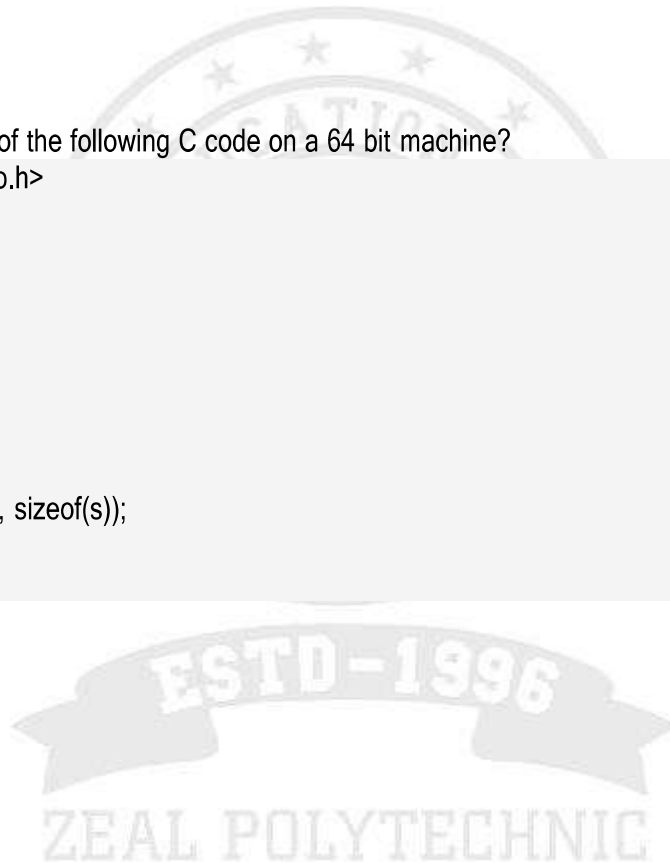
Answer: b

37. What will be the output of the following C code on a 64 bit machine?

```
1. #include <stdio.h>
2. union Sti
3. {
4.     int nu;
5.     char m;6.
6. };
7. int main()
8. {
9.     union Sti s;
10.    printf("%d", sizeof(s));
11.    return 0;
12. }
```

- a) 8
- b) 5
- c) 9
- d) 4

Answer: d





38. What will be the output of the following C function?

```
1. #include <stdio.h>
2. enum birds {SPARROW, PEACOCK, PARROT};
3. enum animals {TIGER = 8, LION, RABBIT, ZEBRA};
4. int main()
5. {
6.     enum birds m = TIGER;
7.     int k;
8.     k = m;
9.     printf("%d\n", k);
10.    return 0;
11. }
```

- a) 0
- b) Compile time error
- c) 1
- d) 8

Answer: d

39. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int const print()
3. {
4.     printf("Sanfoundry.com");
5.     return 0;6.
6. }
7. void main()
8. {
9.     print();
10. }
```

- a) Error because function name cannot be preceded by const
- b) Sanfoundry.com
- c) Sanfoundry.com is printed infinite times
- d) Blank screen, no output

Answer: b

40. Will the following C code compile without any error?

```
1. #include <stdio.h>
2. int main()
3. {
4.     for (int k = 0; k < 10; k++);
5.     return 0;
6. }
```

- a) Yes
- b) No
- c) Depends on the C standard implemented by compilers
- d) Error

Answer: c

41. What will be the final value of x in the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int x = 5 * 9 / 3 + 9;
5. }
```

- a) 3.75
- b) Depends on compiler
- c) 24
- d) 3

Answer: c

42. What will be the output of the following C code? (Initial values: x= 7, y = 8)

```
1. #include <stdio.h>
2. void main()
3. {
4.     float x;
5.     int y;
6.     printf("enter two numbers \n", x);
7.     scanf("%f %f", &x, &y);
8.     printf("%f, %d", x, y);
9. }
```

- a) 7.000000, 7
- b) Run time error
- c) 7.000000, junk
- d) Varies

Answer: c

43. What will be the output of the following C code considering the size of a short int is 2, char is 1 and int is 4 bytes?

```
1. #include <stdio.h>
2. int main()
3. {
4.     short int i = 20;
5.     char c = 97;
6.     printf("%d, %d, %d\n", sizeof(i), sizeof(c), sizeof(c + i));
7.     return 0;
8. }
```

- a) 2, 1, 2
- b) 2, 1, 1
- c) 2, 1, 4
- d) 2, 2, 8

Answer: c

44. What is the difference between the following 2 C codes?

```
1. #include <stdio.h> //Program 1
2. int main()
3. {
4.     int d, a = 1, b = 2;
5.     d = a++ + ++b;
6.     printf("%d %d %d", d, a, b);
7. }
1. #include <stdio.h> //Program 2
2. int main()
3. {
4.     int d, a = 1, b = 2;
5.     d = a++ +++b;
6.     printf("%d %d %d", d, a, b);
7. }
```

- a) No difference as space doesn't make any difference, values of a, b, d are same in both the case
- b) Space does make a difference, values of a, b, d are different
- c) Program 1 has syntax error, program 2 is not
- d) Program 2 has syntax error, program 1 is not

Answer: d

45. What will be the output of the following C code snippet?

```
1. #include <stdio.h>
2. void main()
3. {
4.     1 < 2 ? return 1: return 2;
5. }
```

- a) returns 1
- b) returns 2
- c) Varies
- d) Compile time error

Answer: d

46. What will be the value of the following assignment expression?

(x = foo())!= 1 considering foo() returns 2

- a) 2
- b) True
- c) 1
- d) 0

Answer: a

47. What will be the output of the following C function?

```
1. #include <stdio.h>
2. void reverse(int i);
3. int main()
4. {
5.     reverse(1);
6. }
7. void reverse(int i)
8. {
9.     if (i > 5)
10.        return ;
11.    printf("%d ", i);
12.    return reverse((i++, i));
13. }
```

- a) 1 2 3 4 5
- b) Segmentation fault
- c) Compilation error
- d) Undefined behaviour

Answer: a

48. What will be the final values of i and j in the following C code?

```
1. #include <stdio.h>
2. int x = 0;
3. int main()
4. {
5.     int i = (f() + g()) | g(); //bitwise or
6.     int j = g() | (f() + g()); //bitwise or
7. }
8. int f()
9. {
10.    if (x == 0)
11.        return x + 1;
12.    else
13.        return x - 1;
14. }
15. int g()
16. {
17.    return x++;
18. }
```

- a) i value is 1 and j value is 1
- b) i value is 0 and j value is 0
- c) i value is 1 and j value is undefined
- d) i and j value are undefined

Answer: c

49. Comment on the following C statement.

```
n = 1;
printf("%d, %dn", 3*n, n++);
```

- a) Output will be 3, 2
- b) Output will be 3, 1
- c) Output will be 6, 1
- d) Output is compiler dependent

Answer: d

50. How many times i value is checked in the following C program?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0;
5.     while (i < 3)
6.         i++;
7.     printf("In while loop\n");
8. }
```

- a) 2
- b) 3
- c) 4
- d) 1

Answer: c

51. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0;
5.     do
6.     {
7.         i++;
8.         if (i == 2)
9.             continue;
10.        printf("In while loop ");
11.    } while (i < 2);
12.    printf("%d\n", i);
13. }
```

- a) In while loop 2
- b) In while loop in while loop 3
- c) In while loop 3
- d) Infinite loop

Answer: a

52. What will be the data type returned for the following C function?

```
1. #include <stdio.h>
2. int func()
3. {
4.     return (double)(char)5.0;
5. }
```

- a) char
- b) int
- c) double
- d) multiple type-casting in return is illegal

Answer: b

53. What is the problem in the following C declarations?

```
int func(int);
double func(int);
int func(float);
```

- a) A function with same name cannot have different signatures
- b) A function with same name cannot have different return types
- c) A function with same name cannot have different number of parameters
- d) All of the mentioned

Answer: d

54. Which option should be selected to work the following C expression?

```
string p = "HELLO";
```

- a) typedef char [] string;
- b) typedef char \*string;
- c) typedef char [] string; and typedef char \*string;
- d) Such expression cannot be generated in C

Answer: b

55. What is the meaning of the following C statement?

```
printf("%10s", state);
```

- a) 10 spaces before the string state is printed
- b) Print empty spaces if the string state is less than 10 characters
- c) Print the last 10 characters of the string
- d) None of the mentioned

Answer: b

56. What are the elements present in the array of the following C code?

```
int array[5] = {5};
```

- a) 5, 5, 5, 5, 5
- b) 5, 0, 0, 0, 0
- c) 5, (garbage), (garbage), (garbage), (garbage)
- d) (garbage), (garbage), (garbage), (garbage), 5

Answer: b

57. What will be the output of the following C function when EOF returns?

```
int fputs(char *line, FILE *fp)
```

- a) '\0' character of array line is encountered
- b) 'n' character in array line is encountered
- c) 't' character in array line is encountered
- d) When an error occurs

Answer: d

58. Which part of the program address space is p stored in the following C code?

```
1. #include <stdio.h>
2. int *p;
3. int main()
4. {
5.     int i = 0;
6.     p = &i;
7.     return 0;
8. }
```

- a) Code/text segment
- b) Data segment
- c) Bss segment
- d) Stack

Answer: c

59. Which of the following sequences are unaccepted in C language?

a)  
#if  
#else  
#endif

b)  
#if  
#elif  
#endif

c)  
#if  
#if  
#endif

d)  
#if  
#undef  
#endif

Answer: c

59. Comment on the output of following C code.

```
1. #include <stdio.h>
2. main()
3. {
4.     char *p = 0;
5.     *p = 'a';
6.     printf("value in pointer p is %c\n", *p);
7. }
```

- a) It will print a
- b) It will print 0
- c) Compile time error
- d) Run time error

Answer:d

60. What is the output of this C code?

```
1. #include <stdio.h>
2. main()
3. {
4.     if (sizeof(int) > -1)
5.         printf("True");
6.     else
7.         printf("False");
8. }
```

- a) True
- b) False

Answer:b

61. What is the output of this C code?

```
1. #include <stdio.h>
2. main()
3. {
4.     char *p = "Sanfoundry C-Test";
5.     p[0] = 'a';
6.     p[1] = 'b';
7.     printf("%s", p);
8. }
```

- a) abnfoundry C-Test
- b) Sanfoundry C-Test
- c) Compile time error
- d) Run time error

Answer: d



62. What is the output of this C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     float f = 0.1;
5.     if (f == 0.1)
6.         printf("True");
7.     else
8.         printf("False");
9. }
```

- a) True
- b) False

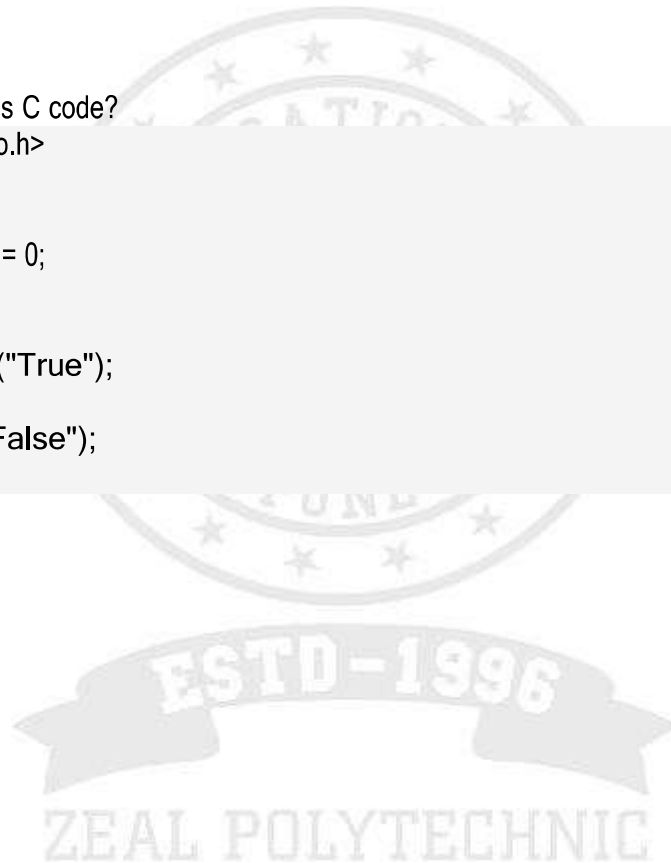
Answer:a

63. What is the output of this C code?

```
1. #include <stdio.h>
2. main()
3. {
4.     int n = 0, m = 0;
5.     if (n > 0)
6.         if (m > 0)
7.             printf("True");
8.     else
9.         printf("False");
10. }
```

- a) True
- b) False
- c) No Output will be printed
- d) Run Time Error

Answer:c



### Unit-3 Control Structures

- 3.1 Decision making logical operators. If statement. If else statement, nested if-else, if-else ladder, the switch statement.
- 3.2 Looping: While loop, Do while loop, For loop, Go to statement, Use of break and continue statements

64. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int x = 5;
5.     if (x < 1)
6.         printf("hello");
7.     if (x == 5)
8.         printf("hi");
9.     else
10.        printf("no");
11. }
```

- a) hi
- b) hello
- c) no
- d) error

Answer: a

65. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int x;
3. void main()
4. {
5.     if (x)
6.         printf("hi");
7.     else
8.         printf("how are u");9.
9. }
```

- a) hi
- b) how are you
- c) compile time error
- d) error

Answer: b

66. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int x = 5;
5.     if (true);
6.     printf("hello");
7. }
```

- a) It will display hello
- b) It will throw an error
- c) Nothing will be displayed
- d) Compiler dependent

Answer: b

67. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int x = 0;
5.     if (x == 0)
6.         printf("hi");
7.     else
8.         printf("how are u");
9.         printf("hello");
10. }
```

- a) hi
- b) how are you
- c) hello
- d) hihello

Answer: d

68. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int x = 5;
5.     if (x < 1);
6.     printf("Hello");
7.
8. }
```

- a) Nothing
- b) Run time error
- c) Hello
- d) Varies

Answer: c

69. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
1. #include <stdio.h>
2. void main()
3. {
4.     double ch;
5.     printf("enter a value between 1 to 2:");
6.     scanf("%lf", &ch);
7.     switch (ch)
8.     {
9.         case 1:
10.            printf("1");
11.            break;
12.         case 2:
13.            printf("2");
14.            break;
15.     }
16. }
```

- a) Compile time error
- b) 1
- c) 2
- d) Varies

Answer: a

70. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
1. #include <stdio.h>
2. void main()
3. {
4.     char *ch;
5.     printf("enter a value between 1 to 3:");
6.     scanf("%s", ch);
7.     switch (ch)
8.     {
9.         case "1":
10.            printf("1");
11.            break;
12.         case "2":
13.            printf("2");
14.            break;
15.     }
16. }
```

- a) 1
- b) 2
- c) Compile time error
- d) No Compile time error

Answer: c

71. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
1. #include <stdio.h>
2. void main()
3. {
4.     int ch;
5.     printf("enter a value between 1 to 2:");
6.     scanf("%d", &ch);
7.     switch (ch)
8.     {
9.         case 1:
10.            printf("1\n");
11.        default:
12.            printf("2\n");
13.    }
14. }
```

- a) 1
- b) 2
- c) 1 2
- d) Run time error

Answer: c

72. What will be the output of the following C code? (Assuming that we have entered the value 2 in the standard input)

```
1. #include <stdio.h>
2. void main()
3. {
4.     int ch;
5.     printf("enter a value between 1 to 2:");
6.     scanf("%d", &ch);
7.     switch (ch)
8.     {
9.         case 1:
10.            printf("1\n");
11.            break;
12.            printf("Hi");
13.        default:
14.            printf("2\n");
15.    }
16. }
```

- a) 1
- b) Hi 2
- c) Run time error
- d) 2

Answer: d

73. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
1. #include <stdio.h>
2. void main()
3. {
4.     int ch;
5.     printf("enter a value between 1 to 2:");
6.     scanf("%d", &ch);
7.     switch (ch, ch + 1)
8.     {
9.         case 1:
10.            printf("1\n");
11.            break;
12.         case 2:
13.            printf("2");
14.            break;
15.     }
16. }
```

- a) 1
- b) 2
- c) 3
- d) Run time error

Answer: b

74. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
1. #include <stdio.h>
2. void main()
3. {
4.     double ch;
5.     printf("enter a value between 1 to 2:");
6.     scanf("%lf", &ch);
7.     switch (ch)
8.     {
9.         case 1:
10.            printf("1");
11.            break;
12.         case 2:
13.            printf("2");
14.            break;
15.     }
16. }
```

- a) Compile time error
- b) 1
- c) 2
- d) Varies

Answer: a

a) What will be the output of the following C code? Assuming that we have entered the value 1 in the standard input

```
1. #include <stdio.h>
2. void main()
3. {
4.     char *ch;
5.     printf("enter a value between 1 to 3:");
6.     scanf("%s", ch);
7.     switch (ch)
8.     {
9.         case "1":
10.            printf("1");
11.            break;
12.         case "2":
13.            printf("2");
14.            break;
15.     }
16. }
```

- a) 1
- b) Compile time error
- c) 2
- d) Run time error

Answer: b

75. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
1. #include <stdio.h>
2. void main()
3. {
4.     int ch;
5.     printf("enter a value between 1 to 2:");
6.     scanf("%d", &ch);
7.     switch (ch)
8.     {
9.         case 1:
10.            printf("1\n");
11.         default:
12.            printf("2\n");
13.     }
14. }
```

- a) 1
- b) 2
- c) 1 2
- d) Run time error

Answer: c

76. What will be the output of the following C code? (Assuming that we have entered the value 2 in the standard input)

```
1. #include <stdio.h>
2. void main()
3. {
4.     int ch;
5.     printf("enter a value between 1 to 2:");
6.     scanf("%d", &ch);
7.     switch (ch)
8.     {
9.         case 1:
10.            printf("1\n");
11.            break;
12.            printf("hi");
13.        default:
14.            printf("2\n");
15.    }
16. }
```

- a) 1
- b) hi 2
- c) Run time error
- d) 2

Answer: d

77. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)

```
1. #include <stdio.h>
2. void main()
3. {
4.     int ch;
5.     printf("enter a value between 1 to 2:");
6.     scanf("%d", &ch);
7.     switch (ch, ch + 1)
8.     {
9.         case 1:
10.            printf("1\n");
11.            break;
12.         case 2:
13.            printf("2");
14.            break;
15.    }
16. }
```

- a) 1
- b) 2
- c) 3
- d) Run time error

Answer: b



78. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int a = 1, b = 1;
5.     switch (a)
6.     {
7.         case a*b:
8.             printf("yes ");
9.         case a-b:
10.            printf("no\n");
11.            break;
12.     }
13. }
```

- a) yes
- b) no
- c) Compile time error
- d) yes no

Answer: c

79. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int x = 97;
5.     switch (x)
6.     {
7.         case 'a':
8.             printf("yes ");
9.             break;
10.        case 97:
11.            printf("no\n");
12.            break;
13.     }
14. }
```

- a) yes
- b) yes no
- c) Duplicate case value error
- d) Character case value error

Answer: c

80. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     float f = 1;
5.     switch (f)
6.     {
7.         case 1.0:
8.             printf("yes\n");
9.             break;
10.        default:
11.            printf("default\n");
12.    }
13. }
```

- a) yes
- b) yes default
- c) Undefined behaviour
- d) Compile time error

Answer: d

81. The C code 'for(;;)' represents an infinite loop. It can be terminated by \_\_\_\_\_

- a) break
- b) exit(0)
- c) abort()
- d) terminate

Answer: a

82. What will be the correct syntax for running two variables for loop simultaneously?

- a)  
for (i = 0; i < n; i++)  
for (j = 0; j < n; j += 5)
- b)  
for (i = 0, j = 0; i < n, j < n; i++, j += 5)
- c)  
for (i = 0; i < n; i++){  
for (j = 0; j < n; j += 5){}
- d) none of the mentioned

Answer: b

83. Which for loop has range of similar indexes of 'i' used in for (i = 0; i < n; i++)?

- a) for (i = n; i > 0; i-)
- b) for (i = n; i >= 0; i-)
- c) for (i = n-1; i > 0; i-)
- d) for (i = n-1; i > -1; i-)

Answer: d

84. Which of the following cannot be used as LHS of the expression in for (exp1;exp2; exp3)?

- a) variable
- b) function
- c) typedef
- d) macros

Answer: d

85. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     short i;
5.     for (i = 1; i >= 0; i++)
6.         printf("%d\n", i);
7.
8. }
```

- a) The control won't fall into the for loop
- b) Numbers will be displayed until the signed limit of short and throw a runtime error
- c) Numbers will be displayed until the signed limit of short and program will successfully terminate
- d) This program will get into an infinite loop and keep printing numbers with no errors

Answer: c

86. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int k = 0;
5.     for (k)
6.         printf("Hello");
7. }
```

- a) Compile time error
- b) hello
- c) Nothing
- d) Varies

Answer: a

87. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int k = 0;
5.     for (k < 3; k++)
6.         printf("Hello");
7. }
```

- a) Compile time error
- b) Hello is printed thrice
- c) Nothing
- d) Varies

Answer: a

88. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     double k = 0;
5.     for (k = 0.0; k < 3.0; k++)
6.         printf("Hello");
7. }
```

- a) Run time error
- b) Hello is printed thrice
- c) Hello is printed twice
- d) Hello is printed infinitely

Answer: b

89. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     while ()
5.         printf("In while loop ");
6.     printf("After loop\n");
7. }
```

- a) In while loop after loop
- b) After loop
- c) Compile time error
- d) Infinite loop

Answer: c

90. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     do
5.         printf("In while loop ");
6.     while (0);
7.     printf("After loop\n");
8. }
```

- a) In while loop
- b) In while loop  
after loop
- c) After loop
- d) Infinite loop

Answer: b

91. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0;
5.     do {
6.         i++;
7.         printf("In while loop\n");
8.     } while (i < 3);
9. }
```

- a) In while loop  
In while loop  
In while loop
- b) In while loop  
In while loop
- c) Depends on the compiler
- d) Compile time error

Answer: a

92. How many times i value is checked in the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0;
5.     do {
6.         i++;
7.         printf("in while loop\n");
8.     } while (i < 3);
9. }
```

- a) 2
- b) 3
- c) 4
- d) 1

Answer: b

93. How many times i value is checked in the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0;
5.     while (i < 3)
6.         i++;
7.     printf("In while loop\n");
8. }
```

- a) 2
- b) 3
- c) 4
- d) 1

Answer: c

94. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int i = 2;
5.     do
6.     {
7.         printf("Hi");
8.     } while (i < 2)9.
    }
```

- a) Compile time error
- b) Hi Hi
- c) Hi
- d) Varies

Answer: a

95. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int i = 0;
5.     while (++i)
6.     {
7.         printf("H");
8.     }
9. }
```

- a) H
- b) H is printed infinite times
- c) Compile time error
- d) Varies

Answer: b

96. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int i = 0;
5.     do
6.     {
7.         printf("Hello");
8.     } while (i != 0);
9. }
```

- a) Nothing
- b) H is printed infinite times
- c) Hello
- d) Run time error

Answer: c

97. Which keyword can be used for coming out of recursion?

- a) break
- b) return
- c) exit
- d) both break and return

Answer: b

98. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int a = 0, i = 0, b;
5.     for (i = 0; i < 5; i++)
6.     {
7.         a++;
8.         continue;
9.     }
10. }
```

- a) 2
- b) 3
- c) 4
- d) 5

Answer: d

99. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int a = 0, i = 0, b;
5.     for (i = 0; i < 5; i++)
6.     {
7.         a++;
8.         if (i == 3)
9.             break;
10.    }
11. }
```

- a) 1
- b) 2
- c) 3
- d) 4

Answer: d

102. The keyword 'break' cannot be simply used within \_\_\_\_\_

- a) do-while
- b) if-else
- c) for
- d) while

Answer: b

103. Which keyword is used to come out of a loop only for that iteration?

- a) break
- b) continue
- c) return
- d) none of the mentioned

Answer: b

104. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int i = 0, j = 0;
5.     for (i = 0; i < 5; i++)
6.     {
7.         for (j = 0; j < 4; j++)
8.         {
9.             if (i > 1)
10.                break;
11.         }
12.         printf("Hi \n");
13.     }
14. }
```

- a) Hi is printed 5 times
- b) Hi is printed 9 times
- c) Hi is printed 7 times
- d) Hi is printed 4 times

Answer: a

105. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int i = 0;
5.     int j = 0;
6.     for (i = 0; i < 5; i++)
7.     {
8.         for (j = 0; j < 4; j++)
9.         {
10.            if (i > 1)
11.                continue;
12.            printf("Hi \n");
13.        }
14.    }
15. }
```

- a) Hi is printed 9 times
- b) Hi is printed 8 times
- c) Hi is printed 7 times
- d) Hi is printed 6 times

Answer: b



106. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int i = 0;
5.     for (i = 0; i < 5; i++)
6.         if (i < 4)
7.             {
8.                 printf("Hello");
9.                 break;
10.            }
11. }
```

- a) Hello is printed 5 times
- b) Hello is printed 4 times
- c) Hello
- d) Hello is printed 3 times

Answer: c

107. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     printf("%d ", 1);
5.     goto l1;
6.     printf("%d ", 2);
7.     l1:goto l2;
8.     printf("%d ", 3);
9.     l2:printf("%d ", 4);
10. }
```

- a) 1 4
- b) Compilation error
- c) 1 2 4
- d) 1 3 4

Answer: a

108. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     printf("%d ", 1);
5.     l1:l2:
6.     printf("%d ", 2);
7.     printf("%d\n", 3);
8. }
```

- a) Compilation error
- b) 1 2 3
- c) 1 2
- d) 1 3

Answer: b

109. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     printf("%d ", 1);
5.     goto l1;
6.     printf("%d ", 2);
7. }
8. void foo()
9. {
10.    l1 : printf("3 ", 3);
11. }
```

- a) 1 2 3
- b) 1 3
- c) 1 3 2
- d) Compilation error

Answer- D

110. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0, j = 0;
5.     while (i < 2)
6.     {
7.         l1 : i++;
8.         while (j < 3)
9.         {
10.            printf("Loop\n");
11.            goto l1;
12.        }
13.    }
14. }
```

- a) Loop Loop
- b) Compilation error
- c) Loop Loop Loop Loop

Answer: d

111. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0, j = 0;
5.     while (l1: i < 2)
6.     {
7.         i++;
8.         while (j < 3)
9.         {
10.            printf("loop\n");
11.            goto l1;
12.        }
13.    }
14. }
```

- a) loop loop
- b) Compilation error
- c) loop loop loop loop
- d) Infinite loop

Answer: b

112. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0, j = 0;
5.     l1: while (i < 2)
6.     {
7.         i++;
8.         while (j < 3)
9.         {
10.            printf("loop\n");
11.            goto l1;
12.        }
13.    }
14. }
```

- a) loop loop
- b) compilation error
- c) oop loop loop loop
- d) infinite loop

Answer: a

113. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int i = 0;
5.     if (i == 0)
6.     {
7.         goto label;
8.     }
9.     label: printf("Hello");
10. }
```

- a) Nothing
- b) Error
- c) Infinite Hello
- d) Hello

Answer: d

114. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int i = 0, k;
5.     if (i == 0)
6.         goto label;
7.     for (k = 0; k < 3; k++)
8.     {
9.         printf("hi\n");
10.        label: k = printf("%03d", i);
11.    }
12. }
```

- a) 0
- b) hi hi hi 0 0 0
- c) 0 hi hi hi 0 0 0
- d) 0 0 0

Answer- A

115. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int i = 0, k;
5.     label: printf("%d", i);
6.     if (i == 0)
7.         goto label;
8. }
```

- a) 0
- b) Infinite 0
- c) Nothing
- d) Error

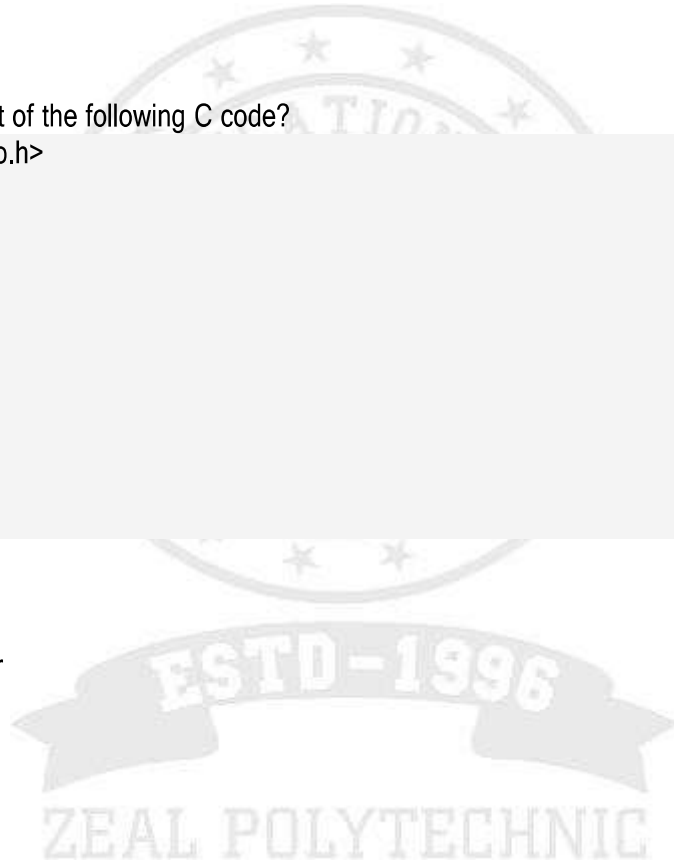
Answer: b

116. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     void foo();
5.     printf("1 ");
6.     foo();7.
7. }
8. void foo()
9. {
10.    printf("2 ");
11. }
```

- a) 1 2
- b) Compile time error
- c) 1 2 1 2
- d) Depends on the compiler

Answer: a



117. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     void foo(), f();
5.     f();
6. }
7. void foo()
8. {
9.     printf("2 ");
10. }
11. void f()
12. {
13.     printf("1 ");
14.     foo();
15. }
```

- a) Compile time error as foo is local to main
- b) 1 2
- c) 2 1
- d) Compile time error due to declaration of functions inside main

Answer: b

118. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     void foo();
5.     void f()
6.     {
7.         foo();
8.     }
9.     f();
10. }
11. void foo()
12. {
13.     printf("2 ");
14. }
```

- a) 2 2
- b) 2
- c) Compile time error
- d) Depends on the compiler

Answer: d

119. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo();
3. int main()
4. {
5.     void foo();
6.     foo();
7.     return 0;8.
8. }
9. void foo()
10. {
11.     printf("2 ");
12. }
```

- a) Compile time error
- b) 2
- c) Depends on the compiler
- d) Depends on the standard

Answer: b

120. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo();
3. int main()
4. {
5.     void foo(int);
6.     foo(1);
7.     return 0;
8. }
9. void foo(int i)
10. {
11.     printf("2 ");
12. }
```

- a) 2
- b) Compile time error
- c) Depends on the compiler
- d) Depends on the standard

Answer: a

121. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo();
3. int main()
4. {
5.     void foo(int);
6.     foo();
7.     return 0;
8. }
9. void foo()
10. {
11.     printf("2 ");
12. }
```

- a) 2
- b) Compile time error
- c) Depends on the compiler
- d) Depends on the standard

Answer: b

122. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void m()
3. {
4.     printf("hi");
5. }
6. void main()
7. {
8.     m();
9. }
```

- a) hi
- b) Run time error
- c) Nothing
- d) Varies

Answer: a



123. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void m();
3. void n()
4. {
5.     m();
6. }
7. void main()
8. {
9.     void m()
10.    {
11.        printf("hi");
12.    }
13. }
```

- a) hi
- b) Compile time error
- c) Nothing
- d) Varies

Answer: b

124. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     m();
5.     void m()
6.     {
7.         printf("hi");
8.     }
9. }
```

- a) hi
- b) Compile time error
- c) Nothing
- d) Varies

Answer: b

125. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     m();
5. }
6. void m()
7. {
8.     printf("hi");
9.     m();
10. }
```

- a) Compile time error
- b) hi
- c) Infinite hi
- d) Nothing

Answer: c

126. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     static int x = 3;
5.     x++;
6.     if (x <= 5)
7.     {
8.         printf("hi");
9.         main();
10.    }
11. }
```

- a) Run time error
- b) hi
- c) Infinite hi
- d) hi hi

Answer: d

127. Which of the following is a correct format for declaration of function?

- a) return-type function-name(argument type);
- b) return-type function-name(argument type){}
- c) return-type (argument type)function-name;
- d) all of the mentioned

Answer: a

128. Which of the following function declaration is illegal?

- a) int 1bhk(int);
- b) int 1bhk(int a);
- c) int 2bhk(int\*, int []);
- d) all of the mentioned

Answer: d

129. Which function definition will run correctly?

a)  
int sum(int a, int b)  
return (a + b);

b)  
int sum(int a, int b)  
{return (a + b);}

c)  
int sum(a, b)  
return (a + b);

- e) none of the mentioned

Answer: b

130. Can we use a function as a parameter of another function? [Eg: void wow(int func())].

- a) Yes, and we can use the function value conveniently
- b) Yes, but we call the function again to get the value, not as convenient as in using variable
- c) No, C does not support it
- d) This case is compiler dependent

Answer: c

131. The value obtained in the function is given back to main by using \_\_\_\_\_ keyword.

- a) return
- b) static
- c) new
- d) volatile

Answer: a

132. What is the return-type of the function sqrt()?

- a) int
- b) float
- c) double
- d) depends on the data type of the parameter

Answer: c

133. Which of the following function declaration is illegal?

a)  
`double func();`  
`int main(){}`  
`double func(){}`

b)  
`double func(){};`  
`int main(){}`

c)  
`int main()`  
`{`  
`double func();`  
`}`  
`double func(){//statements}`

d) None of the mentioned

Answer: d

134. What will be the output of the following C code having void return-type function?

```
1. #include <stdio.h>
2. void foo()
3. {
4.     return 1;
5. }
6. void main()
7. {
8.     int x = 0;
9.     x = foo();
10.    printf("%d", x);
11. }
```

- a) 1
- b) 0
- c) Runtime error
- d) Compile time error

Answer: d

135. What will be the data type returned for the following C function?

```
1. #include <stdio.h>
2. int func()
3. {
4.     return (double)(char)5.0;
5. }
```

- a) char
- b) int
- c) double
- d) multiple type-casting in return is illegal

Answer: b

136. What is the problem in the following C declarations?

```
int func(int);
double func(int);
int func(float);
```

- a) A function with same name cannot have different signatures
- b) A function with same name cannot have different return types
- c) A function with same name cannot have different number of parameters
- d) All of the mentioned

Answer: d

137. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int m()
3. {
4.     printf("hello");
5. }
6. void main()
7. {
8.     int k = m();
9.     printf("%d", k);
10. }
```

- a) hello5
- b) Error
- c) Nothing
- d) Junk value

Answer: a

138. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int *m()
3. {
4.     int *p = 5;
5.     return p;6.
6. }
7. void main()
8. {
9.     int *k = m();
10.    printf("%d", k);
11. }
```

- a) 5
- b) Junk value
- c) 0
- d) Error

Answer: a

139. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int *m();
3. void main()
4. {
5.     int *k = m();
6.     printf("hello ");
7.     printf("%d", k[0]);
8. }
9. int *m()
10. {
11.     int a[2] = {5, 8};
12.     return a;
13. }
```

- a) hello 5 8
- b) hello 5
- c) hello followed by garbage value
- d) Compilation error

Answer: c

140. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     m();
5.     printf("%d", x);
6. }
7. int x;
8. void m()
9. {
10.     x = 4;
11. }
```

- a) 4
- b) Compile time error
- c) 0
- d) Undefined

Answer: b

141. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int x;
3. void main()
4. {
5.     printf("%d", x);
6. }
```

- a) Junk value
- b) Run time error
- c) 0
- d) Undefined

Answer: c

142. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int x = 5;
3. void main()
4. {
5.     int x = 3;
6.     printf("%d", x);
7.     {
8.         x = 4;
9.     }
10.    printf("%d", x);
11. }
```

- a) Run time error
- b) 3 3
- c) 3 5
- d) 3 4

Answer: d

143. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int x = 5;
3. void main()
4. {
5.     int x = 3;
6.     printf("%d", x);
7.     {
8.         int x = 4;
9.     }
10.    printf("%d", x);
11. }
```

- a) 3 3
- b) 3 4
- c) 3 5
- d) Run time error

Answer: a

144. Functions in C are always \_\_\_\_\_

- a) Internal

- b) External
- c) Both Internal and External
- d) External and Internal are not valid terms for functions

Answer: b

145. Global variables are \_\_\_\_\_

- a) Internal
- b) External
- c) Both Internal and External
- d) None of the mentioned

Answer: b

146. Which of the following is an external variable in the following C code?

```
1. #include <stdio.h>
2. int func (int a)
3. {
4.     int b;
5.     return b;6.
6. }
7. int main()
8. {
9.     int c;
10.    func (c);
11. }
12. int d;
```

- a) a
- b) b
- c) c
- d) d

Answer: d

147. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     printf("%d", d++);
5. }
6. int d = 10;
```

- a) 9
- b) 10
- c) 11
- d) Compile time error

Answer: d



148. What will be the output of the following C code?

```
1. #include <stdio.h>
2. double var = 8;
3. int main()
4. {
5.     int var = 5;
6.     printf("%d", var);
7. }
```

- a) 5
- b) 8
- c) Compile time error due to wrong format identifier for double
- d) Compile time error due to redeclaration of variable with same name

Answer: a

149. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int i;
3. int main()
4. {
5.     extern int i;
6.     if (i == 0)
7.         printf("scope rules\n");
8. }
```

- a) scope rules
- b) Compile time error due to multiple declaration
- c) Compile time error due to not defining type in statement extern i
- d) Nothing will be printed as value of i is not zero because i is an automatic variable

Answer: a

150. What will be the output of the following C code (without linking the source file in which ary1 is defined)?

```
1. #include <stdio.h>
2. int main()
3. {
4.     extern ary1[];
5.     printf("scope rules\n");
6. }
```

- a) scope rules
- b) Linking error due to undefined reference
- c) Compile time error because size of array is not provided
- d) Compile time error because datatype of array is not provided

Answer: a

151. What will be the output of the following C code (after linking to source file having definition of ary1)?

```
1. #include <stdio.h>
2. int main()
3. {
4.     extern ary1[];
5.     printf("%d\n", ary1[0]);
6. }
```

- a) Value of ary1[0];
- b) Compile time error due to multiple definition
- c) Compile time error because size of array is not provided
- d) Compile time error because datatype of array is not provided

Answer: d

152. What is the scope of an external variable?

- a) Whole source file in which it is defined
- b) From the point of declaration to the end of the file in which it is defined
- c) Any source file in a program
- d) From the point of declaration to the end of the file being compiled

Answer: d

153. What is the scope of a function?

- a) Whole source file in which it is defined
- b) From the point of declaration to the end of the file in which it is defined
- c) Any source file in a program
- d) From the point of declaration to the end of the file being compiled

Answer: d

154. Comment on the output of the following C code.

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i;
5.     for (i = 0; i < 5; i++)
6.         int a = i;
7.     printf("%d", a);
8. }
```

- a) a is out of scope when printf is called
- b) Redeclaration of a in same scope throws error
- c) Syntax error in declaration of a
- d) No errors, program will show the output 5

Answer: c.

155. Which variable has the longest scope in the following C code?

```
1. #include <stdio.h>
2. int b;
3. int main()
4. {
5.     int c;
6.     return 0;
7. }
8. int a;
```

- a) a
- b) b
- c) c
- d) Both a and b

Answer: b

156. Comment on the following 2 C programs.

```
1. #include <stdio.h> //Program 1
2. int main()
3. {
4.     int a;
5.     int b;
6.     int c;
7. }
8.
9. #include <stdio.h> //Program 2
10. int main()
11. {
12.     int a;
13.     {
14.         int b;
15.     }
16.     {
17.         int c;
18.     }
19. }
```

- a) Both are same
- b) Scope of c is till the end of the main function in Program 2
- c) In Program 1, variables a, b and c can be used anywhere in the main function whereas in Program 2, variables b and c can be used only inside their respective blocks
- d) None of the mentioned

Answer: c

157. What will be the output of the following C code if these two files namely test.c and test1.c are linked and run?

```
1.  -----file test.c-----
2.  #include <stdio.h>
3.  #include ""test.h""
4.  int main()
5.  {
6.      i = 10;
7.      printf(""%d """, i);
8.      foo();
9.  }
10.
11.  ----file test1.c-----
12.  #include <stdio.h>
13.  #include ""test.h""
14.  int foo()
15.  {
16.      printf(""%d\n""", i);
17.  }
18.
19.  ----file test.h-----
20.  #include <stdio.h>
21.  #include <stdlib.h>
22.  static int i;
```

- a) 10 0
- b) 0 0
- c) 10 10
- d) Compilation Error

Answer: a

158. Functions have static qualifier for its declaration by default.

- a) True
- b) False
- c) Depends on the compiler
- d) Depends on the standard

Answer: b

159. Is initialisation mandatory for local static variables?

- a) Yes
- b) No
- c) Depends on the compiler
- d) Depends on the standard

Answer: b

160. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     foo();
5.     foo();
6. }
7. void foo()
8. {
9.     int i = 11;
10.    printf("%d ", i);
11.    static int j = 12;
12.    j = j + 1;
13.    printf("%d\n", j);
14. }
```

- a) 11 12 11 12
- b) 11 13 11 14
- c) 11 12 11 13
- d) Compile time error

Answer: b

161. Assignment statements assigning value to local static variables are executed only once.

- a) True
- b) False
- c) Depends on the code
- d) None of the mentioned

Answer: b

162. What is the format identifier for "static a = 20.5;"?

- a) %s
- b) %d
- c) %f
- d) Illegal declaration due to absence of data type

Answer: b

163. Which of the following is true for the static variable?

- a) It can be called from another function
- b) It exists even after the function ends
- c) It can be modified in another function by sending it as a parameter
- d) All of the mentioned

Answer: b.

164. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void func();
3. int main()
4. {
5.     static int b = 20;
6.     func();
7. }
8. void func()
9. {
10.     static int b;
11.     printf("%d", b);
12. }
```

- a) Output will be 0
- b) Output will be 20
- c) Output will be a garbage value
- d) Compile time error due to redeclaration of static variable

Answer: a

165. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     register int i = 10;
5.     int *p = &i;
6.     *p = 11;
7.     printf("%d %d\n", i, *p);
8. }
```

- a) Depends on whether i is actually stored in machine register
- b) 10 10
- c) 11 11
- d) Compile time error

Answer: d

166. register keyword mandates compiler to place it in machine register.

- a) True
- b) False
- c) Depends on the standard
- d) None of the mentioned

Answer: b

167. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     register static int i = 10;
5.     i = 11;
6.     printf("%d\n", i);
7. }
```

- a) 10
- b) Compile time error
- c) Undefined behaviour
- d) 11

Answer: b

168. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     register auto int i = 10;
5.     i = 11;
6.     printf("%d\n", i);
7. }
```

- a) 10
- b) Compile time error
- c) Undefined behaviour
- d) 11

Answer: b

169. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     register const int i = 10;
5.     i = 11;
6.     printf("%d\n", i);
7. }
```

- a) 10
- b) Compile time error
- c) Undefined behaviour
- d) 11

Answer: b

170. Register storage class can be specified to global variables.

- a) True
- b) False
- c) Depends on the compiler
- d) Depends on the standard

Answer: b

171. Which among the following is wrong for "register int a;"?

- a) Compiler generally ignores the request
- b) You cannot take the address of this variable
- c) Access time to a is critical
- d) None of the mentioned

Answer: d

172. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     register int x = 5;
5.     m();
6.     printf("x is %d", x);
7. }
8. void m()
9. {
10.    x++;
11. }
```

- a) 6
- b) 5
- c) Junk value
- d) Compile time error

Answer: d

173. What is the scope of an automatic variable?

- a) Within the block it appears
- b) Within the blocks of the block it appears
- c) Until the end of program
- d) Within the block it appears & Within the blocks of the block it appears

Answer: d

174. Automatic variables are allocated space in the form of a \_\_\_\_\_

- a) stack
- b) queue
- c) priority queue
- d) random

Answer: a

175. Which of the following is a storage specifier?

- a) enum
- b) union
- c) auto
- d) volatile

Answer: c



176. If storage class is not specified for a local variable, then the default class will be auto.

- a) True
- b) False
- c) Depends on the standard
- d) None of the mentioned

Answer: a

177. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo(auto int i);
3. int main()
4. {
5.     foo(10);
6. }
7. void foo(auto int i)
8. {
9.     printf("%d\n", i);
10. }
```

- a) 10
- b) Compile time error
- c) Depends on the standard
- d) None of the mentioned

Answer: b

178. Automatic variables are stored in \_\_\_\_\_

- a) stack
- b) data segment
- c) register
- d) heap

Answer: a

179. What linkage does automatic variables have?

- a) Internal linkage
- b) External linkage
- c) No linkage
- d) None of the mentioned

Answer: c

180. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     auto i = 10;
5.     const auto int *p = &i;
6.     printf("%d\n", i);
7. }
```

- a) 10
- b) Compile time error
- c) Depends on the standard
- d) Depends on the compiler

Answer: a

181. Property which allows to produce different executable for different platforms in C is called?

- a) File inclusion
- b) Selective inclusion
- c) Conditional compilation
- d) Recursive macros

Answer: c

182. What is #include <stdio.h>?

- a) Preprocessor directive
- b) Inclusion directive
- c) File inclusion directive
- d) None of the mentioned

Answer: a

183. C preprocessors can have compiler specific features.

- a) True
- b) False
- c) Depends on the standard
- d) Depends on the platform

Answer: a

184. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define foo(m, n) m * n = 10
3. int main()
4. {
5.     printf("in main\n");
6. }
```

- a) In main
- b) Compilation error as lvalue is required for the expression m\*n=10
- c) Preprocessor error as lvalue is required for the expression m\*n=10
- d) None of the mentioned

Answer: a

185. C preprocessor is conceptually the first step during compilation.

- a) True
- b) False
- c) Depends on the compiler
- d) Depends on the standard

Answer: a

186. Preprocessor feature that supply line numbers and filenames to compiler is called?

- a) Selective inclusion
- b) macro substitution
- c) Concatenation
- d) Line control

Answer: d

187. #include <somefile.h> are \_\_\_\_\_ files and #include "somefile.h" \_\_\_\_\_ files.

- a) Library, Library
- b) Library, user-created header
- c) User-created header, library
- d) They can include all types of file

Answer: d

188. What is a preprocessor?

- a) That processes its input data to produce output that is used as input to another program
- b) That is nothing but a loader
- c) That links various source files
- d) All of the mentioned

Answer: a

189. Which of the following are C preprocessors?

- a) #ifdef
- b) #define
- c) #endif
- d) all of the mentioned

Answer: d

190. #include statement must be written \_\_\_\_\_

- a) Before main()
- b) Before any scanf/printf
- c) After main()
- d) It can be written anywhere

Answer: a

191. #pragma exit is primarily used for?

- a) Checking memory leaks after exiting the program
- b) Informing Operating System that program has terminated
- c) Running a function at exiting the program
- d) No such preprocessor exist

Answer: c

192. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int one = 1, two = 2;
5.     #ifdef next
6.     one = 2;
7.     two = 1;
8.     #endif
9.     printf("%d, %d", one, two);
10. }
```

- a) 1, 1
- b) 1, 2
- c) 2, 1
- d) 2, 2

Answer: b

193. The C-preprocessors are specified with \_\_\_\_\_ symbol.

- a) #
- b) \$
- c) " "
- d) &

Answer: a

194. What is #include directive?

- a) Tells the preprocessor to grab the text of a file and place it directly into the current file
- b) Statements are not typically placed at the top of a program
- c) All of the mentioned
- d) None of the mentioned

Answer: a

195. The preprocessor provides the ability for \_\_\_\_\_

- a) The inclusion of header files
- b) The inclusion of macro expansions
- c) Conditional compilation and line control
- d) All of the mentioned

Answer: d

196. If #include is used with file name in angular brackets.

- a) The file is searched for in the standard compiler include paths
- b) The search path is expanded to include the current source directory
- c) The search path will expand
- d) None of the mentioned

Answer: a

197. What is the sequence for preprocessor to look for the file within <>?

- a) The predefined location then the current directory
- b) The current directory then the predefined location
- c) The predefined location only
- d) The current directory location

Answer: a

198. Which directory the compiler first looks for the file when using #include?

- a) Current directory where program is saved
- b) C:COMPILERINCLUDE
- c) S:SOURCEHEADERS
- d) Both C:COMPILERINCLUDE and S:SOURCEHEADERS simultaneously

Answer: b

199. What would happen if you create a file stdio.h and use #include "stdio.h"?

- a) The predefined library file will be selected
- b) The user-defined library file will be selected
- c) Both the files will be included
- d) The compiler won't accept the program

Answer: b

200. How is search done in #include and #include "somelibrary.h" according to C standard?

- a) When former is used, current directory is searched and when latter is used, standard directory is searched
- b) When former is used, standard directory is searched and when latter is used, current directory is searched
- c) When former is used, search is done in implementation defined manner and when latter is used, current directory is searched
- d) For both, search for 'somelibrary' is done in implementation-defined places

Answer: d

201. How is search done in #include and #include "somelibrary.h" normally or conventionally?
- When former is used, current directory is searched and when latter is used, standard directory is searched
  - When former is used, predefined directory is searched and when latter is used, current directory is searched and then predefined directories are searched
  - When former is used, search is done in implementation defined manner and latter is used to search current directory
  - For both, search for somelibrary is done in implementation-defined manner

Answer: b

202. Can function definition be present in header files?

- Yes
- No
- Depends on the compiler
- Depends on the standard

Answer: a

203. Comment on the output of the following C code.

```
1. #include <stdio.h>
2. #include "test.h"
3. #include "test.h"
4. int main()
5. {
6.     //some code
7. }
```

- True
- Compile time error
- False
- Depends on the compiler

Answer: b

204. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define foo(m, n) m ## n
3. void myfunc();
4. int main()
5. {
6.     myfunc();
7. }
8. void myfunc()
9. {
10.    printf("%d\n", foo(2, 3));
11. }
```

- 23
- 2 3
- Compile time error
- Undefined behaviour

Answer: a

205. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define foo(m, n) m ## n
3. int main()
4. {
5.     printf("%s\n", foo(k, l));
6. }
```

- a) k l
- b) kl
- c) Compile time error
- d) Undefined behaviour

Answer: c

206. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define foo(m, n) " m ## n "
3. int main()
4. {
5.     printf("%s\n", foo(k, l));
6. }
```

- a) k l
- b) kl
- c) Compile time error
- d) m ## n

Answer: d

207. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define foo(x, y) #x #y
3. int main()
4. {
5.     printf("%s\n", foo(k, l));
6.     return 0;
7. }
```

- a) kl
- b) k l
- c) xy
- d) Compile time error

Answer: a

208. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define foo(x, y) x / y + x
3. int main()
4. {
5.     int i = -6, j = 3;
6.     printf("%d\n", foo(i + j, 3));
7.     return 0;
8. }
```

- a) Divided by zero exception
- b) Compile time error
- c) -8
- d) -4

Answer: c

209. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void f();
3. int main()
4. {
5.     #define foo(x, y) x / y + x
6.     f();
7. }
8. void f()
9. {
10.     printf("%d\n", foo(-3, 3));
11. }
```

- a) -8
- b) -4
- c) Compile time error
- d) Undefined behaviour

Answer: b

210. What will be the output of the following C code?



```

1. #include <stdio.h>
2. void f();
3. int main()
4. {
5.     #define max 10
6.     f();
7.     return 0;
8. }
9. void f()
10. {
11.     printf("%d\n", max * 10);
12. }

```

- a) 100
- b) Compile time error since #define cannot be inside functions
- c) Compile time error since max is not visible in f()
- d) Undefined behaviour

Answer: a

211. What will be the output of the following C code?

```

1. #include <stdio.h>
2. #define foo(x, y) x / y + x
3. int main()
4. {
5.     int i = -6, j = 3;
6.     printf("%d ", foo(i + j, 3));
7.     printf("%d\n", foo(-3, 3));
8.     return 0;
9. }

```

- a) -8 -4
- b) -4 divided by zero exception
- c) -4 -4
- d) Divided by zero exception

Answer: a

212. What will be the output of the following C code?

```

1. #include <stdio.h>
2. int foo(int, int);
3. #define foo(x, y) x / y + x
4. int main()
5. {
6.     int i = -6, j = 3;
7.     printf("%d ",foo(i + j, 3));
8.     #undef foo
9.     printf("%d\n",foo(i + j, 3));
10. }
11. int foo(int x, int y)
12. {
13.     return x / y + x;
14. }

```

- a) -8 -4

- b) Compile time error
- c) -8 -8
- d) Undefined behaviour

Answer: a

213. What is the advantage of #define over const?

- a) Data type is flexible
- b) Can have a pointer
- c) Reduction in the size of the program
- d) None of the mentioned

Answer: a

214. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define SYSTEM 20
3. int main()
4. {
5.     int a = 20;
6.     #if SYSTEM == a
7.     printf("HELLO ");
8.     #endif
9.     #if SYSTEM == 20
10.    printf("WORLD\n");
11.    #endif
12. }
```

- a) HELLO
- b) WORLD
- c) HELLO WORLD
- d) No Output

Answer: b

215. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define Cprog
3. int main()
4. {
5.     int a = 2;
6.     #ifdef Cprog
7.     a = 1;
8.     printf("%d", Cprog);
9. }
```

- a) No output on execution
- b) Output as 1
- c) Output as 2
- d) Compile time error

Answer: d

216. The "else if" in conditional inclusion is written by?

- a) #else if
- b) #elseif
- c) #elsif
- d) #elif

Answer: d.

217. What will be the output of the following C code?

```
1. #include <stdio.h>
2. #define COLD
3. int main()
4. {
5.     #ifdef COLD
6.     printf("COLD\t");
7.     #undef COLD
8.     #endif
9.     #ifdef COLD
10.    printf("HOT\t");
11.    #endif
12. }
```

- a) HOT
- b) COLD
- c) COLD HOT
- d) No Output

Answer: b

218. Which of the following sequences are unaccepted in C language?

a)  
#if  
#else  
#endif

b)  
#if  
#elif  
#endif

c)  
#if  
#if  
#endif

d)  
#if  
#undef  
#endif

Answer: c

219. In a conditional inclusion, if the condition that comes after the if is true, then what will happen during

compilation?

- a) Then the code up to the following #else or #elif or #endif is compiled
- b) Then the code up to the following #endif is compiled even if #else or #elif is present
- c) Then the code up to the following #elif is compiled
- d) None of the mentioned

Answer: a

220. Conditional inclusion can be used for \_\_\_\_\_

- a) Preventing multiple declarations of a variable
- b) Check for existence of a variable and doing something if it exists
- c) Preventing multiple declarations of same function
- d) All of the mentioned

Answer: d

221. The #elif directive cannot appear after the preprocessor #else directive.

- a) True
- b) False

Answer: a



Unit-4 Array and Structure
----------------------------

- |   |
|---|
| <ul style="list-style-type: none"><li>4.1 Characteristics of an array. One dimension and two dimensions arrays</li><li>4.2 Array declaring and Initialization</li><li>4.3 Array of characters, Operation on array</li><li>4.4 Character and string input/output</li><li>4.5 Introduction and Features of structures, Declaration and Initialization</li><li>4.6 Type def, Enumerated Data Type using structure in C program</li></ul> |
|---|

222. What will be the output of the following C code?

```

1. #include <stdio.h>
2. int main()
3. {
4.     char *p = NULL;
5.     char *q = 0;
6.     if (p)
7.         printf(" p ");
8.     else
9.         printf("nullp");
10.    if (q)
11.        printf("q\n");
12.    else
13.        printf(" nullq\n");
14. }

```

- a) nullp nullq
- b) Depends on the compiler
- c) x nullq where x can be p or nullp depending on the value of NULL
- d) p q

Answer: a

223. What will be the output of the following C code?

```

1. #include <stdio.h>
2. int main()
3. {
4.     int i = 10;
5.     void *p = &i;
6.     printf("%d\n", (int)*p);
7.     return 0;8.
}

```

- a) Compile time error
- b) Segmentation fault/runtime crash
- c) 10
- d) Undefined behaviour

Answer: a

224. What will be the output of the following C code?

```

1. #include <stdio.h>
2. int main()
3. {
4.     int i = 10;
5.     void *p = &i;
6.     printf("%f\n", *(float*)p);
7.     return 0;8.
}

```

- a) Compile time error
- b) Undefined behaviour
- c) 10
- d) 0.000000

Answer: d

225. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int *f();
3. int main()
4. {
5.     int *p = f();
6.     printf("%d\n", *p);
7. }
8. int *f()
9. {
10.    int *j = (int*)malloc(sizeof(int));
11.    *j = 10;
12.    return j;
13. }
```

- a) 10
- b) Compile time error
- c) Segmentation fault/runtime crash since pointer to local variable is returned
- d) Undefined behaviour

Answer: a

226. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int *f();
3. int main()
4. {
5.     int *p = f();
6.     printf("%d\n", *p);
7. }
8. int *f()
9. {
10.    int j = 10;
11.    return &j;
12. }
```

- a) 10
- b) Compile time error
- c) Segmentation fault/runtime crash
- d) Undefined behaviour

Answer: a

227. Comment on the following pointer declaration.

`int *ptr, p;`

- a) ptr is a pointer to integer, p is not
- b) ptr and p, both are pointers to integer
- c) ptr is a pointer to integer, p may or may not be
- d) ptr and p both are not pointers to integer

Answer: a

228. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int *ptr, a = 10;
5.     ptr = &a;
6.     *ptr += 1;
7.     printf("%d,%d/n", *ptr, a);
8. }
```

- a) 10,10
- b) 10,11
- c) 11,10
- d) 11,11

Answer: d

229. Comment on the following C statement.

```
const int *ptr;
```

- a) You cannot change the value pointed by ptr
- b) You cannot change the pointer ptr itself
- c) You May or may not change the value pointed by ptr
- d) You can change the pointer as well as the value pointed by it

Answer: a

230. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo(int*);
3. int main()
4. {
5.     int i = 10;
6.     foo(&i);
7. }
8. void foo(int *p)
9. {
10.    printf("%d\n", *p);
11. }
```

- a) 10
- b) Some garbage value
- c) Compile time error
- d) Segmentation fault/code crash

Answer: c

231. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo(int*);
3. int main()
4. {
5.     int i = 10, *p = &i;
6.     foo(p++);
7. }
8. void foo(int *p)
9. {
10.    printf("%d\n", *p);
11. }
```

- a) 10
- b) Some garbage value
- c) Compile time error
- d) Segmentation fault

Answer: a

232. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo(float *);
3. int main()
4. {
5.     int i = 10, *p = &i;
6.     foo(&i);
7. }
8. void foo(float *p)
9. {
10.    printf("%f\n", *p);
11. }
```

- a) 10.000000
- b) 0.000000
- c) Compile time error
- d) Undefined behaviour

Answer: b



233. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 97, *p = &i;
5.     foo(&i);
6.     printf("%d ", *p);
7. }
8. void foo(int *p)
9. {
10.    int j = 2;
11.    p = &j;
12.    printf("%d ", *p);
13. }
```

- a) 2 97
- b) 2 2
- c) Compile time error
- d) Segmentation fault/code crash

Answer: a

234. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 97, *p = &i;
5.     foo(&p);
6.     printf("%d ", *p);
7.     return 0;
8. }
9. void foo(int **p)
10. {
11.    int j = 2;
12.    *p = &j;
13.    printf("%d ", **p);
14. }
```

- a) 2 2
- b) 2 97
- c) Undefined behaviour
- d) Segmentation fault/code crash

Answer: a

235. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 11;
5.     int *p = &i;
6.     foo(&p);
7.     printf("%d ", *p);8.
8. }
9. void foo(int *const *p)
10. {
11.     int j = 10;
12.     *p = &j;
13.     printf("%d ", **p);
14. }
```

- a) Compile time error
- b) 10 10
- c) Undefined behaviour
- d) 10 11

Answer: a

236. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 10;
5.     int *p = &i;
6.     foo(&p);
7.     printf("%d ", *p);
8.     printf("%d ", *p);9.
9. }
10. void foo(int **const p)
11. {
12.     int j = 11;
13.     *p = &j;
14.     printf("%d ", **p);
15. }
```

- a) 11 11 11
- b) 11 11 Undefined-value
- c) Compile time error
- d) Segmentation fault/code-crash

Answer: b

237. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 10;
5.     int *const p = &i;
6.     foo(&p);
7.     printf("%d\n", *p);
8. }
9. void foo(int **p)
10. {
11.     int j = 11;
12.     *p = &j;
13.     printf("%d\n", **p);
14. }
```

- a) 11 11
- b) Undefined behaviour
- c) Compile time error
- d) Segmentation fault/code-crash

Answer: a

238. Which of the following is the correct syntax to send an array as a parameter to function?

- a) func(&array);
- b) func(#array);
- c) func(\*array);
- d) func(array[size]);

Answer: a

239. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     int a[3] = {1, 2, 3};
5.     int *p = a;
6.     printf("%p\t%p", p, a);
7. }
```

- a) Same address is printed
- b) Different address is printed
- c) Compile time error
- d) Nothing

Answer: a

240. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     char *s = "hello";
5.     char *p = s;
6.     printf("%p\t%p", p, s);
7. }
```

- a) Different address is printed
- b) Same address is printed
- c) Run time error
- d) Nothing

Answer: b

241. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     char *s= "hello";
5.     char *p = s;
6.     printf("%c\t%c", p[0], s[1]);
7. }
```

- a) Run time error
- b) h h
- c) h e
- d) h l

Answer: c

242. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     char *s= "hello";
5.     char *p = s;
6.     printf("%c\t%c", *(p + 3), s[1]);
7. }
```

- a) h e
- b) l l
- c) l o
- d) l e

Answer: d

243. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     char *s= "hello";
5.     char *p = s;
6.     printf("%c\t%c", 1[p], s[1]);
7. }
```

- a) h h
- b) Run time error
- c) ll
- d) e e

244. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo( int[] );
3. int main()
4. {
5.     int ary[4] = {1, 2, 3, 4};
6.     foo(ary);
7.     printf("%d ", ary[0]);
8. }
9. void foo(int p[4])
10. {
11.     int i = 10;
12.     p = &i;
13.     printf("%d ", p[0]);
14. }
```

- a) 10 10
- b) Compile time error
- c) 10 1
- d) Undefined behaviour

Answer: c

245. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int ary[4] = {1, 2, 3, 4};
5.     int *p = ary + 3;
6.     printf("%d\n", p[-2]);
7. }
```

- a) 1
- b) 2
- c) Compile time error
- d) Some garbage value

Answer: b

246. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int ary[4] = {1, 2, 3, 4};
5.     int *p = ary + 3;
6.     printf("%d %d\n", p[-2], ary[*p]);
7. }
```

- a) 2 3
- b) Compile time error
- c) 2 4
- d) 2 somegarbagevalue

247. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     double *ptr = (double *)100;
5.     ptr = ptr + 2;
6.     printf("%u", ptr);
7. }
```

- a) 102
- b) 104
- c) 108
- d) 116

Answer: d

248. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int *p = (int *)2;
5.     int *q = (int *)3;
6.     printf("%d", p + q);
7. }
```

- a) 2
- b) 3
- c) 5
- d) Compile time error

Answer: d

249. Which of the following arithmetic operation can be applied to pointers a and b?  
(Assuming initialization as `int *a = (int *)2; int *b = (int *)3;`)

- a) `a + b`
- b) `a - b`
- c) `a * b`
- d) `a / b`

Answer: b

4. What is the size of `*ptr` in a 32-bit machine (Assuming initialization as `int *ptr = 10;`)?

- a) 1
- b) 2
- c) 4
- d) 8

Answer: c

250. Which of following logical operation can be applied to pointers?

(Assuming initialization `int *a = 2; int *b = 3;`)

- a) `a | b`
- b) `a ^ b`
- c) `a & b`
- d) None of the mentioned

Answer: d

251. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     char *s = "hello";
5.     char *p = s;
6.     printf("%c\t%c", *(p + 1), s[1]);
7. }
```

- a) h e
- b) e l
- c) h h
- d) e e

Answer: d

252. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     char *s = "hello";
5.     char *p = s;
6.     printf("%c\t%c", *p, s[1]);
7. }
```

- a) e h
- b) Compile time error
- c) h h
- d) h e

Answer: d

253. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     char *s = "hello";
5.     char *n = "cjn";
6.     char *p = s + n;
7.     printf("%c\t%c", *p, s[1]);
8. }
```

- a) h e
- b) Compile time error
- c) c o
- d) h n

Answer: b

254. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char *str = "hello, world\n";
5.     char *strc = "good morning\n";
6.     strcpy(strc, str);
7.     printf("%s\n", strc);
8.     return 0;
9. }
```

- a) hello, world
- b) Crash/segmentation fault
- c) Undefined behaviour
- d) Run time error

Answer: b

255. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char *str = "hello world";
5.     char strc[] = "good morning india\n";
6.     strcpy(strc, str);
7.     printf("%s\n", strc);
8.     return 0;
9. }
```

- a) hello world
- b) hello worldg india
- c) Compile time error
- d) Undefined behaviour

Answer: a

256. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char *str = "hello, world!!\n";
5.     char strc[] = "good morning\n";
6.     strcpy(strc, str);
7.     printf("%s\n", strc);
8.     return 0;
9. }
```

- a) hello, world!!
- b) Compile time error
- c) Undefined behaviour
- d) Segmentation fault

Answer: c



257. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char *str = "hello, world\n";
5.     str[5] = '!';
6.     printf("%s\n", str);
7.     return 0;
8. }
```

- a) hello. world
- b) hello, world
- c) Compile error
- d) Segmentation fault

Answer: d

258. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char str[] = "hello, world";
5.     str[5] = '!';
6.     printf("%s\n", str);
7.     return 0;8.
}
```

- a) hello. world
- b) hello, world
- c) Compile error
- d) Segmentation fault

Answer: a

259. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char *str = "hello world";
5.     char strary[] = "hello world";
6.     printf("%d %d\n", sizeof(str), sizeof(strary));
7.     return 0;8.
}
```

- a) 11 11
- b) 12 12
- c) 4 12
- d) 4 11

Answer: c

260. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char *str = "hello world";
5.     char strary[] = "hello world";
6.     printf("%d %d\n", strlen(str), strlen(strary));
7.     return 0;
8. }
```

- a) 11 11
- b) 12 11
- c) 11 12
- d) x 11 where x can be any positive integer.

Answer: a

261. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void f(char *k)
3. {
4.     k++;
5.     k[2] = 'm';
6.     printf("%c\n", *k);
7. }
8. void main()
9. {
10.     char s[] = "hello";
11.     f(s);
12. }
```

- a) l
- b) e
- c) h
- d) o

Answer: b

262. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void fun(char *k)
3. {
4.     printf("%s", k);
5. }
6. void main()
7. {
8.     char s[] = "hello";
9.     fun(s);
10. }
```

- a) hello
- b) Run time error
- c) Nothing
- d) h

Answer: a

263. What is the correct syntax to send a 3-dimensional array as a parameter? (Assuming declaration int

a[5][4][3];)

- a) func(a);
- b) func(&a);
- c) func(\*a);
- d) func(\*\*a);

Answer: a

264. What are the applications of a multidimensional array?

- a) Matrix-Multiplication
- b) Minimum Spanning Tree
- c) Finding connectivity between nodes
- d) All of the mentioned

Answer: d

265. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo(int *ary[]);
3. int main()
4. {
5.     int ary[2][3];
6.     foo(ary);
7. }
8. void foo(int *ary[])
9. {
10.    int i = 10, j = 2, k;
11.    ary[0] = &i;
12.    ary[1] = &j;
13.    *ary[0] = 2;
14.    for (k = 0; k < 2; k++)
15.        printf("%d\n", *ary[k]);
16. }
```

- a) 2 2
- b) Compile time error
- c) Undefined behaviour
- d) 10 2

Answer: a

266. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void foo(int (*ary)[3]);
3. int main()
4. {
5.     int ary[2][3];
6.     foo(ary);
7. }
8. void foo(int (*ary)[3])
9. {
10.    int i = 10, j = 2, k;
11.    ary[0] = &i;
12.    ary[1] = &j;
13.    for (k = 0; k < 2; k++)
14.        printf("%d\n", *ary[k]);
15. }
```

- a) Compile time error
- b) 10 2
- c) Undefined behaviour
- d) segmentation fault/code crash

Answer- A

267. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     foo(ary);
5. }
6. void foo(int **ary)
7. {
8.     int i = 10, k = 20, j = 30;
9.     int *ary[2];
10.    ary[0] = &i;
11.    ary[1] = &j;
12.    printf("%d\n", ary[0][1]);
13. }
```

- a) 10
- b) 20
- c) Compile time error
- d) Undefined behaviour

Answer: d

268. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int ary[2][3][4], j = 20;
5.     ary[0][0] = &j;
6.     printf("%d\n", *ary[0][0]);
7. }
```

- a) Compile time error
- b) 20
- c) Address of j
- d) Undefined behaviour

Answer: a

269. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int ary[2][3];
5.     ary[] = {{1, 2, 3}, {4, 5, 6}};
6.     printf("%d\n", ary[1][0]);
7. }
```

- a) Compile time error
- b) 4
- c) 1
- d) 2

Answer: a

270. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char *p[1] = {"hello"};
5.     printf("%s", (p)[0]);
6.     return 0;
7. }
```

- a) Compile time error
- b) Undefined behaviour
- c) hello
- d) None of the mentioned

Answer: c

271. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char **p = {"hello", "hi", "bye"};
5.     printf("%s", (p)[0]);
6.     return 0;
7. }
```

- a) Compile time error
- b) Undefined behaviour
- c) hello
- d) Address of hello

Answer: b

272. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0, j = 1;
5.     int *a[] = {&i, &j};
6.     printf("%d", (*a)[0]);
7.     return 0;
8. }
```

- a) Compile time error
- b) Undefined behaviour
- c) 0
- d) Some garbage value

Answer: c

273. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0, j = 1;
5.     int *a[] = {&i, &j};
6.     printf("%d", *a[0]);
7.     return 0;
8. }
```

- a) Compile time error
- b) Undefined behaviour
- c) 0
- d) Some garbage value

Answer: c

274. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     int i = 0, j = 1;
5.     int *a[] = {&i, &j};
6.     printf("%d", (*a)[1]);
7.     return 0;
8. }
```

- a) Compile time error
- b) Undefined behaviour
- c) 1
- d) Some garbage value

Answer: d

275. Which of the following are generated from char pointer?

- a) char \*string = "Hello.;"
- b) char \*string;  
scanf("%s", string);
- c) char string[] = "Hello.;"
- d) char \*string = "Hello.;" and char string[] = "Hello.;"

Answer: a

276. Which of the following declaration are illegal?

- a) int a[][] = {{1, 2, 3}, {2, 3, 4, 5}};
- b) int \*a[] = {{1, 2, 3}, {2, 3, 4, 5}};
- c) int a[4][4] = {{1, 2, 3}, {2, 3, 4, 5}};
- d) none of the mentioned

Answer: a

277. What will be the output of the following C code (considering sizeof char is 1 and pointer is 4)?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char *a[2] = {"hello", "hi"};
5.     printf("%d", sizeof(a));
6.     return 0;
7. }
```

- a) 9
- b) 4
- c) 8
- d) 10

Answer: c

278. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char a[2][6] = {"hello", "hi"};
5.     printf("%d", sizeof(a));
6.     return 0;
7. }
```

- a) 9
- b) 12
- c) 8
- d) 10

Answer: b

279. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char a[2][6] = {"hello", "hi"};
5.     printf("%s", *a + 1);
6.     return 0;
7. }
```

- a) hello
- b) hi
- c) ello
- d) ello hi

Answer: c

280. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     char *a[2] = {"hello", "hi"};
5.     printf("%s", *(a + 1));
6.     return 0;
7. }
```

- a) hello
- b) ello
- c) hi
- d) ello hi

Answer: c

281. What is the advantage of a multidimensional array over pointer array?

- a) Predefined size
- b) Input can be taken from user
- c) Faster Access
- d) All of the mentioned

Answer: d



282. Which of the following operation is possible using a pointer char? (Assuming the declaration is char \*a;)

- a) Input via %s
- b) Generation of the multidimensional array
- c) Changing address to point at another location
- d) All of the mentioned

Answer: c

283. Comment on the following two operations.

```
int *a[] = {{1, 2, 3}, {1, 2, 3, 4}}; //- 1  
int b[4][4] = {{1, 2, 3}, {1, 2, 3, 4}}; //- 2
```

- a) 1 will work, 2 will not
- b) 1 and 2, both will work
- c) 1 won't work, 2 will work
- d) Neither of them will work

Answer: c

284. Comment on the following two operations.

```
int *a[] = {{1, 2, 3}, {1, 2, 3, 4}}; //- 1  
int b[][] = {{1, 2, 3}, {1, 2, 3, 4}}; //- 2
```

- a) 1 works, 2 doesn't
- b) 2 works, 1 doesn't
- c) Both of them work
- d) Neither of them work

Answer: d

285. A program that has no command line arguments will have argc \_\_\_\_\_

- a) Zero
- b) Negative
- c) One
- d) Two

Answer: c

286. What is the index of the last argument in command line arguments?

- a) argc - 2
- b) argc + 1
- c) argc
- d) argc - 1

Answer: d

287. What will be the output of the following C code (if run with no options or arguments)?

```
1. #include <stdio.h>
2. int main(int argc, char *argv[])
3. {
4.     printf("%d\n", argc);
5.     return 0;6.
   }
```

- a) 0
- b) 1
- c) Depends on the platform
- d) Depends on the compiler

Answer: b

288. What will be the output of the following C code (run without any command line arguments)?

```
1. #include <stdio.h>
2. int main(int argc, char *argv[])
3. {
4.     while (argc--)
5.         printf("%s\n", argv[argc]);
6.     return 0;7.
   }
```

- a) Compile time error
- b) Executable filename
- c) Segmentation fault
- d) Undefined

Answer: b

289. What will be the output of the following C code (run without any command line arguments)?

```
1. #include <stdio.h>
2. int main(int argc, char *argv[])
3. {
4.     printf("%s\n", argv[argc]);
5.     return 0;6.
   }
```

- a) Segmentation fault/code crash
- b) Executable file name
- c) Depends on the platform
- d) Depends on the compiler

Answer: a

290. What will be the output of the following C code (run without any command line arguments)?

```
1. #include <stdio.h>
2. int main(int argc, char *argv[])
3. {
4.     while (*argv++ != NULL)
5.         printf("%s\n", *argv);
6.     return 0;
7. }
```

- a) Segmentation fault/code crash
- b) Executable file name
- c) Depends on the platform
- d) Depends on the compiler

Answer: a

291. What will be the output of the following C code (run without any command line arguments)?

```
1. #include <stdio.h>
2. int main(int argc, char *argv[])
3. {
4.     while (*argv != NULL)
5.         printf("%s\n", *(argv++));
6.     return 0;
7. }
```

- a) Segmentation fault/code crash
- b) Executable file name
- c) Depends on the platform
- d) Depends on the compiler

Answer: b

292. What will be the output of the following C code (run without any command line arguments)?

```
1. #include <stdio.h>
2. int main(int argc, char *argv[])
3. {
4.     while (argv != NULL)
5.         printf("%s\n", *(argv++));
6.     return 0;
7. }
```

- a) Segmentation fault/code crash
- b) Executable file name
- c) Depends on the platform
- d) Depends on the compiler

Answer: a

## Unit-5 Functions

5.1 Concept and need of functions

5.2 Library function: Math functions, String handling functions, other miscellaneous functions.

5.3 Writing user defined functions, scope of variables.

5.4 Parameter passing: call by value, call by reference.

5.5 Recursive functions.

293. What will be the output of the following C code?

```
1. #include <stdio.h>
2. void main()
3. {
4.     struct student
5.     {
6.         int no;
7.         char name[20];
8.     };
9.     struct student s;
10.    no = 8;
11.    printf("%d", no);
12. }
```

- a) Nothing
- b) Compile time error
- c) Junk
- d) 8

Answer: b

294. How many bytes in memory taken by the following C structure?

```
1. #include <stdio.h>
2. struct test
3. {
4.     int k;
5.     char c;
6. };
```

- a) Multiple of integer size
- b) integer size+character size
- c) Depends on the platform
- d) Multiple of word size

Answer: a

295. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct
3. {
4.     int k;
5.     char c;
6. };
7. int main()
8. {
9.     struct p;
10.    p.k = 10;
11.    printf("%d\n", p.k);
12. }
```

- a) Compile time error
- b) 10
- c) Undefined behaviour
- d) Segmentation fault

Answer: a

296. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct
3. {
4.     int k;
5.     char c;
6. } p;
7. int p = 10;
8. int main()
9. {
10.    p.k = 10;
11.    printf("%d %d\n", p.k, p);
12. }
```

- a) Compile time error
- b) 10 10
- c) Depends on the standard
- d) Depends on the compiler

Answer: a

297. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct p
3. {
4.     int k;
5.     char c;
6. };
7. int p = 10;
8. int main()
9. {
10.     struct p x;
11.     x.k = 10;
12.     printf("%d %d\n", x.k, p);
13. }
```

- a) Compile time error
- b) 10 10
- c) Depends on the standard
- d) Depends on the compiler

Answer: b

298. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct p
3. {
4.     int k;
5.     char c;
6.     float f;
7. };
8. int p = 10;
9. int main()
10. {
11.     struct p x = {1, 97};
12.     printf("%f %d\n", x.f, p);
13. }
```

- a) Compile time error
- b) 0.000000 10
- c) Somegarbage value 10
- d) 0 10

Answer: b

299. What will be the output of the following C code according to C99 standard?

```
1. #include <stdio.h>
2. struct p
3. {
4.     int k;
5.     char c;
6.     float f;7.
7. };
8. int main()
9. {
10.     struct p x = {.c = 97, .f = 3, .k = 1};
11.     printf("%f\n", x.f);
12. }
```

- a) 3.000000
- b) Compile time error
- c) Undefined behaviour
- d) 1.000000

Answer: a

300. What will be the output of the following C code according to C99 standard?

```
1. #include <stdio.h>
2. struct p
3. {
4.     int k;
5.     char c;
6.     float f;7.
7. };
8. int main()
9. {
10.     struct p x = {.c = 97, .k = 1, 3};
11.     printf("%f \n", x.f);
12. }
```

- a) 3.000000
- b) 0.000000
- c) Compile time error
- d) Undefined behaviour

Answer: b

301. What will be the output of the following C code according to C99 standard?

```
1. #include <stdio.h>
2. struct p
3. {
4.     int k;
5.     char c;
6.     float f;
7. };
8. int main()
9. {
10.     struct p x = {.c = 97};
11.     printf("%f\n", x.f);
12. }
```

- a) 0.000000
- b) Somegarbagevalue
- c) Compile time error
- d) None of the mentioned

Answer: a

302. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. int main()
8. {
9.     struct point p = {1};
10.    struct point p1 = {1};
11.    if(p == p1)
12.        printf("equal\n");
13.    else
14.        printf("not equal\n");
15. }
```

- a) Compile time error
- b) equal
- c) depends on the standard
- d) not equal

Answer: a

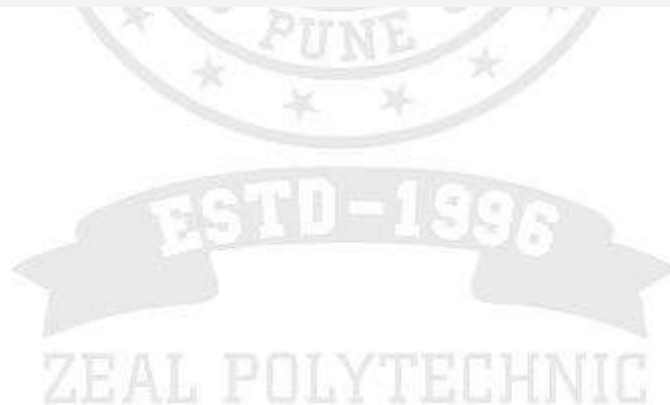


303. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. struct notpoint
8. {
9.     int x;
10.    int y;
11. };
12. struct point foo();
13. int main()
14. {
15.     struct point p = {1};
16.     struct notpoint p1 = {2, 3};
17.     p1 = foo();
18.     printf("%d\n", p1.x);
19. }
20. struct point foo()
21. {
22.     struct point temp = {1, 2};
23.     return temp;
24. }
```

- a) Compile time error
- b) 1
- c) 2
- d) Undefined behaviour

Answer: a



304. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. struct notpoint
8. {
9.     int x;
10.    int y;
11. };
12. int main()
13. {
14.    struct point p = {1};
15.    struct notpoint p1 = p;
16.    printf("%d\n", p1.x);
17. }
```

- a) Compile time error
- b) 1
- c) 0
- d) Undefined

Answer: a

305. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. struct notpoint
8. {
9.     int x;
10.    int y;
11. };
12. void foo(struct point);
13. int main()
14. {
15.    struct notpoint p1 = {1, 2};
16.    foo(p1);
17. }
18. void foo(struct point p)
19. {
20.    printf("%d\n", p.x);
21. }
```

- a) Compile time error
- b) 1
- c) 0
- d) Undefined

Answer: a

306. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. void foo(struct point*);
8. int main()
9. {
10.     struct point p1 = {1, 2};
11.     foo(&p1);
12. }
13. void foo(struct point *p)
14. {
15.     printf("%d\n", *p.x++);
16. }
```

- a) Compile time error
- b) Segmentation fault/code crash
- c) 2
- d) 1

Answer: a

307. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. void foo(struct point*);
8. int main()
9. {
10.     struct point p1 = {1, 2};
11.     foo(&p1);
12. }
13. void foo(struct point *p)
14. {
15.     printf("%d\n", *p->x++);
16. }
```

- a) Compile time error
- b) 1
- c) Segmentation fault/code crash
- d) 2

Answer: a

308. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct student fun(void)
3. {
4.     struct student
5.     {
6.         char *name;
7.     };
8.     struct student s;
9.     s.name = "alan";
10.    return s;
11. }
12. void main()
13. {
14.     struct student m = fun();
15.     printf("%s", m.name);
16. }
```

- a) Compile time error
- b) alan
- c) Nothing
- d) Varies

Answer: a

309. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct student
3. {
4.     char *name;
5. };
6. struct student fun(void)
7. {
8.     struct student s;
9.     s.name = "alan";
10.    return s;
11. }
12. void main()
13. {
14.     struct student m = fun();
15.     printf("%s", m.name);
16. }
```

- a) Nothing
- b) alan
- c) Run time error
- d) Varies

Answer: b

310. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. void foo(struct point*);
8. int main()
9. {
10.     struct point p1[] = {1, 2, 3, 4};
11.     foo(p1);
12. }
13. void foo(struct point p[])
14. {
15.     printf("%d\n", p[1].x);
16. }
```

- a) Compile time error
- b) 3
- c) 2
- d) 1

Answer: b

311. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. void foo(struct point*);
8. int main()
9. {
10.     struct point p1[] = {1, 2, 3, 4};
11.     foo(p1);
12. }
13. void foo(struct point p[])
14. {
15.     printf("%d\n", p->x);
16. }
```

- a) 1
- b) 2
- c) 3
- d) Compile time error

Answer: a.

312. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. void foo(struct point*);
8. int main()
9. {
10.     struct point p1[] = {1, 2, 3, 4};
11.     foo(p1);
12. }
13. void foo(struct point p[])
14. {
15.     printf("%d %d\n", p->x, ++p->x);
16. }
```

- a) 1 2
- b) 2 2
- c) Compile time error
- d) Undefined behaviour

Answer: b

313. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. } p[] = {1, 2, 3, 4, 5};
7. void foo(struct point*);
8. int main()
9. {
10.     foo(p);
11. }
12. void foo(struct point p[])
13. {
14.     printf("%d %d\n", p->x, p[2].y);
15. }
```

- a) 1 0
- b) Compile time error
- c) 1 somegarbagevalue
- d) Undefined behaviour

Answer: a

314. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. void foo(struct point*);
8. int main()
9. {
10.     struct point p1[] = {1, 2, 3, 4, 5};
11.     foo(p1);
12. }
13. void foo(struct point p[])
14. {
15.     printf("%d %d\n", p->x, p[3].y);
16. }
```

- a) Compile time error
- b) 1 0
- c) 1 somegarbagevalue
- d) None of the mentioned

Answer: c

315. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. void foo(struct point*);
8. int main()
9. {
10.     struct point p1[] = {1, 2, 3, 4, 5};
11.     foo(p1);
12. }
13. void foo(struct point p[])
14. {
15.     printf("%d %d\n", p->x, (p + 2).y);
16. }
```

- a) Compile time error
- b) 1 0
- c) 1 somegarbagevalue
- d) Undefined behaviour

Answer: a

316. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct point
3. {
4.     int x;
5.     int y;
6. };
7. void foo(struct point*);
8. int main()
9. {
10.     struct point p1[] = {1, 2, 3, 4, 5};
11.     foo(p1);
12. }
13. void foo(struct point p[])
14. {
15.     printf("%d %d\n", p->x, (p + 2)->y);
16. }
```

- a) Compile time error
- b) 1 0
- c) 1 somegarbagevalue
- d) undefined behaviour

Answer: b

317. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct student
3. {
4.     char *c;
5. };
6. void main()
7. {
8.     struct student s[2];
9.     printf("%d", sizeof(s));
10. }
```

- a) 2
- b) 4
- c) 16
- d) 8

Answer: d



318. What will be the output of the following C

```
1. #include <stdio.h>
2. struct student
3. {
4.     char *c;
5. };
6. void main()
7. {
8.     struct student m;
9.     struct student *s = &m;
10.    s->c = "hello";
11.    printf("%s", s->c);
code? 12. }
```

- a) hello
- b) Run time error
- c) Nothing
- d) Depends on compiler

Answer: a

319. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct student
3. {
4.     char *c;
5. };
6. void main()
7. {
8.     struct student *s;
9.     s->c = "hello";
10.    printf("%s", s->c);
11. }
```

- a) hello
- b) Segmentation fault
- c) Run time error
- d) Nothing

Answer: b

320. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct student
3. {
4.     char *c;
5. };
6. void main()
7. {
8.     struct student m;
9.     struct student *s = &m;
10.    s->c = "hello";
11.    printf("%s", m.c);
12. }
```

- a) Run time error
- b) Nothing
- c) hello
- d) Varies

Answer: c

321. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct student
3. {
4.     char *c;
5. };
6. void main()
7. {
8.     struct student m;
9.     struct student *s = &m;
10.    (*s).c = "hello";
11.    printf("%s", m.c);
12. }
```

- a) Run time error
- b) Nothing
- c) Varies
- d) hello

Answer: d

322. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct student
3. {
4.     char *c;
5. };
6. void main()
7. {
8.     struct student n;
9.     struct student *s = &n;
10.    (*s).c = "hello";
11.    printf("%p\n%p\n", s, &n);
12. }
```

- a) Different address
- b) Run time error
- c) Nothing
- d) Same address

Answer: d.

323. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct p
3. {
4.     int x[2];
5. };
6. struct q
7. {
8.     int *x;
9. };
10. int main()
11. {
12.     struct p p1 = {1, 2};
13.     struct q *ptr1;
14.     ptr1->x = (struct q*)&p1.x;
15.     printf("%d\n", ptr1->x[1]);
16. }
```

- a) Compile time error
- b) Segmentation fault/code crash
- c) 2
- d) 1

324. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct p
3. {
4.     int x[2];
5. };
6. struct q
7. {
8.     int *x;
9. };
10. int main()
11. {
12.     struct p p1 = {1, 2};
13.     struct q *ptr1 = (struct q*)&p1;
14.     ptr1->x = (struct q*)&p1.x;
15.     printf("%d\n", ptr1->x[0]);
16. }
```

- a) Compile time error
- b) Undefined behaviour
- c) Segmentation fault/code crash
- d) 1

Answer: b

325. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct p
3. {
4.     int x;
5.     int y;
6. };
7. int main()
8. {
9.     struct p p1[] = {1, 2, 3, 4, 5, 6};
10.    struct p *ptr1 = p1;
11.    printf("%d %d\n", ptr1->x, (ptr1 + 2)->x);
12. }
```

- a) 1 5
- b) 1 3
- c) Compile time error
- d) 1 4

Answer: a

326. What will be the output of the following C code?

```
1. #include <stdio.h>
2. struct p
3. {
4.     int x;
5.     char y;
6. };
7. int main()
8. {
9.     struct p p1[] = {1, 92, 3, 94, 5, 96};
10.    struct p *ptr1 = p1;
11.    int x = (sizeof(p1) / sizeof(struct p));
12.    printf("%d %d\n", ptr1->x, (ptr1 + x - 1)->x);
13. }
```

- a) Compile time error
- b) Undefined behaviour
- c) 1 3
- d) 1 5

Answer: d

327. What will be the output of the following C code?

```
1. #include <stdio.h>
2. typedef struct p *q;
3. int main()
4. {
5.     struct p
6.     {
7.         int x;
8.         char y;
9.         q ptr;
10.    };
11.    struct p p = {1, 2, &p};
12.    printf("%d\n", p.ptr->x);
13.    return 0;
14. }
```

- a) Compile time error
- b) 1
- c) Depends on the compiler
- d) None of the mentioned

Answer: a

328. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int main()
3. {
4.     typedef struct p *q;
5.     struct p
6.     {
7.         int x;
8.         char y;
9.         q ptr;
10.    };
11.    struct p p = {1, 2, &p};
12.    printf("%d\n", p.ptr->x);
13.    return 0;
14. }
```

- a) Compile time error
- b) 1
- c) Depends on the compiler
- d) Depends on the standard

Answer: b

329. What will be the output of the following C code?

```
1. #include <stdio.h>
2. typedef struct p *q;
3. struct p
4. {
5.     int x;
6.     char y;
7.     q ptr;8.
9. };
9. int main()
10. {
11.     struct p p = {1, 2, &p};
12.     printf("%d\n", p.ptr->ptr->x);
13.     return 0;
14. }
```

- a) Compile time error
- b) Segmentation fault
- c) Undefined behaviour
- d) 1

Answer: d

330. The number of distinct nodes the following struct declaration can point to is \_\_\_\_\_

```
1. struct node
2. {
3.     struct node *left;
4.     struct node *centre;
5.     struct node *right;
6. };
```

- a) 1
- b) 2
- c) 3
- d) All of the mentioned

Answer: d

331. Which of the following is not possible regarding the structure variable?

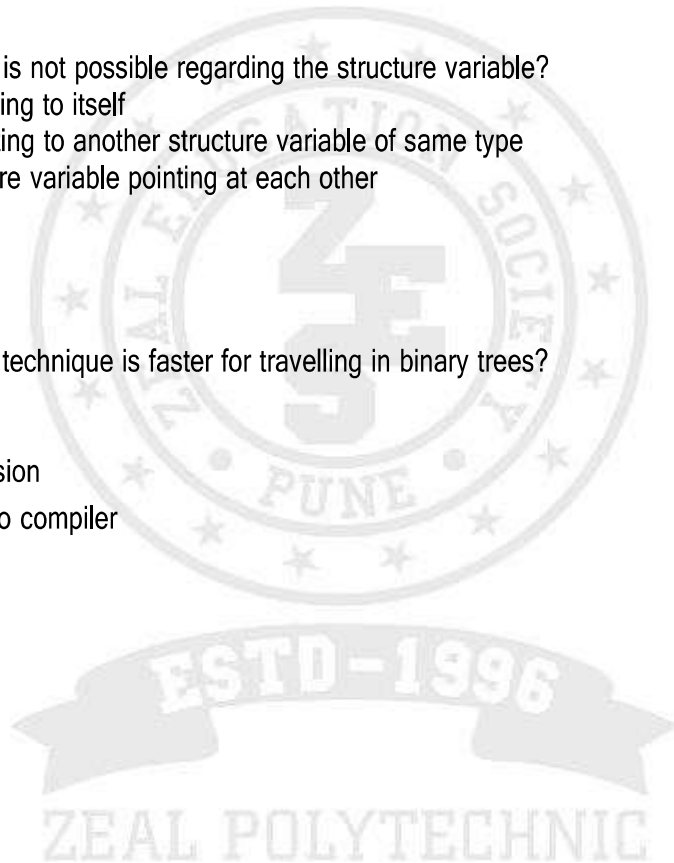
- a) A structure variable pointing to itself
- b) A structure variable pointing to another structure variable of same type
- c) 2 different type of structure variable pointing at each other
- d) None of the mentioned

Answer: d

332. Which of the following technique is faster for travelling in binary trees?

- a) Iteration
- b) Recursion
- c) Both Iteration and Recursion
- d) Depends from compiler to compiler

Answer: b



333. Which of the following will stop the loop at the last node of a linked list in the following C code snippet?

```
1. struct node
2. {
3.     struct node *next;
4. };
```

a)

```
while (p != NULL)
{
    p = p->next;
}
```

b)

```
while (p->next != NULL)
{
    p = p->next;
}
```

c)

```
while (1)
{
    p = p->next;
    if (p == NULL)
        break;
}
```

d) All of the mentioned

Answer: b

334. What will be the output of the following C code?

```
1. #include <stdio.h>
2. typedef struct student
3. {
4.     char *a;
5. }stu;
6. void main()
7. {
8.     struct stu s;
9.     s.a = "hi";
10.    printf("%s", s.a);
11. }
```

a) Compile time error

b) Varies

c) hi

d) h

Answer: a



335. What will be the output of the following C code?

```
1. #include <stdio.h>
2. typedef struct student
3. {
4.     char *a;
5. }stu;
6. void main()
7. {
8.     struct student s;
9.     s.a = "hey";
10.    printf("%s", s.a);
11. }
```

- a) Compile time error
- b) Varies
- c) he
- d) hey

Answer: d

336. What will be the output of the following C code?

```
1. #include <stdio.h>
2. typedef int integer;
3. int main()
4. {
5.     int i = 10, *ptr;
6.     float f = 20;
7.     integer j = i;
8.     ptr = &j;
9.     printf("%d\n", *ptr);
10.    return 0;
11. }
```

- a) Compile time error
- b) Undefined behaviour
- c) Depends on the standard
- d) 10

Answer: d

337. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int (*(x()))[2];
3. typedef int (**ptr())[2] ptrfoo;
4. int main()
5. {
6.     ptrfoo ptr1;
7.     ptr1 = x;
8.     ptr1();
9.     return 0;
10. }
11. int *(x())[2]
12. {
13.     int (*ary)[2] = malloc(sizeof*ary);
14.     return &ary;
15. }
```

- a) Compile time error
- b) Nothing
- c) Undefined behaviour
- d) Depends on the standard

Answer: a

338. What will be the output of the following C code?

```
1. #include <stdio.h>
2. int (*(x()))[2];
3. typedef int (**ptrfoo())[2];
4. int main()
5. {
6.     ptrfoo ptr1;
7.     ptr1 = x;
8.     ptr1();
9.     return 0;
10. }
11. int *(x())[2]
12. {
13.     int (*ary)[2] = malloc(sizeof * ary);
14.     return &ary;
15. }
```

- a) Compile time error
- b) Nothing
- c) Undefined behaviour
- d) Depends on the standard

Answer: b

339. What will be the output of the following C code?

```
1. #include <stdio.h>
2. typedef struct p
3. {
4.     int x, y;
5. };
6. int main()
7. {
8.     p k1 = {1, 2};
9.     printf("%d\n", k1.x);
10. }
```

- a) Compile time error
- b) 1
- c) 0
- d) Depends on the standard

Answer: a

340. What will be the output of the following C code?

```
1. #include <stdio.h>
2. typedef struct p
3. {
4.     int x, y;
5. }k = {1, 2};
6. int main()
7. {
8.     p k1 = k;
9.     printf("%d\n", k1.x);
10. }
```

- a) Compile time error
- b) 1
- c) 0
- d) Depends on the standard

Answer: a

341. What will be the output of the following C code?

```
1. #include <stdio.h>
2. typedef struct p
3. {
4.     int x, y;
5. }k;
6. int main()
7. {
8.     struct p p = {1, 2};
9.     k k1 = p;
10.    printf("%d\n", k1.x);
11. }
```

- a) Compile time error
- b) 1
- c) 0
- d) Depends on the standard

Answer: b



## Unit-6 Pointers

- 6.1 Concepts of pointers: declaring, initializing, accessing.
- 6.2 Pointer arithmetic
- 6.3 Handling arrays using pointers.
- 6.4 Handling functions using pointers.
- 6.5 Handling structure using pointers

342. The preprocessor directive used to give additional information to the compiler, beyond which is conveyed in the language \_\_\_\_\_

- a) #include
- b) #define
- c) #pragma
- d) #elif

Answer: c

343. What will be the output of the following C code, if it is run on a 32 bit platform?

```
#include<stdio.h>
#pragma(1)
struct test
{
    int i;
    char j;
};
main()
{
    printf("%d",sizeof(struct test));
}
```

- a) Error
- b) 1
- c) 4
- d) 8

Answer: d

344. In the directive, #pragma pack(n), which of the following is not a valid value of n?

- a) 1
- b) 2
- c) 3
- d) 4

Answer: c

345. Which of the following attributes is used to specify that the minimum required memory to be used to represent the types?

- a) packed
- b) aligned
- c) unused
- d) deprecated

Answer: a

346. In the directive #pragma pack(n), if the value of 'n' is given to be 5, then what happens?

- a) Error
- b) Warning but no error
- c) Executes the pragma statement
- d) Ignores the pragma statement and executes the program

Answer: d

347. The correct syntax of the attribute packed is \_\_\_\_\_

- a) \_\_attribute\_\_((packed));
- b) \_attribute(packed);
- c) \_attribute\_((packed));
- d) \_\_attribute\_(packed);

Answer: a

348. The pragma \_\_\_\_\_ is used to remove an identifier completely from a program.

- a) GNU piston
- b) GCC poison
- c) GNU poison
- d) GCC piston

Answer: b

349. The function of \_\_attribute\_\_((packed)); can also be performed using \_\_\_\_\_

- a) #pragma pack(1);
- b) #pragma pack(2);
- c) #pragma pack(4);
- d) #pragma pack(8);

Answer: a

350. #pragma GCC poison should be followed by a list of identifiers that are \_\_\_\_\_

- a) even in number
- b) odd in number
- c) valid
- d) invalid

Answer: d

351. What will be the output of the following C code?

```
#include<stdio.h>
#pragma GCC poison printf
main()
{
    printf("sanfoundry");
    return 0;
}
```

- a) error is thrown
- b) sanfoundry is printed
- c) warning but no error
- d) yrdnoufnas is printed

Answer: a

352. Which of the following is a stringizing operator?

- a) < >
- b) #
- c) %
- d) ##

Answer: b

353. What will be the output of the following C code?

```
#define sanfoundry(s,n) #s #n
main()
{
    printf(sanfoundry(hello,world));
}
```

- a) sanfoundry(hello,world)
- b) sanfoundry
- c) hello,world
- d) helloworld

Answer: d

354. What will be the output of the following C code?

```
#define display(text) printf(#text "@")
main()
{
    display(hello.);
    display(good morning!);
}
```

- a) hello.@good morning!
- b) error
- c) hello.good morning!@
- d) hello.@good morning!@

Answer: d

355. What will be the output of the following C code?

```
#define display(a) #a
main()
{
    printf(display("56#7"));
}
```

- a) Error
- b) "56#7"
- c) 56#7
- d) 567

Answer: b

356. What will be the output of the following C code?

```
#define HELLO(a) #a
main()
{
    printf(HELLO(good    morning));
}
```

- a) good morning
- b) goodmorning
- c) good morning
- d) error

Answer: c

357. What will be the output of the following C code?

```
#include <stdio.h>
#define sanfoundry(x) #x
int main()
{
    int marks=100;
    printf("value of %s is = %d\n",sanfoundry(marks),marks);
    return 0;
}
```

- a) error
- b) value of marks=100
- c) value of=100
- d) 100

Answer: b

358. What will be the output of the following C code?

```
#define hello(c) #c
main()
{
    printf(hello(i,am));
}
```

- a) i,am
- b) iam
- c) i am
- d) error

Answer: d

359. What will be the output of the following C code?

```
#define hello(c,d) #c #d
main()
{
    printf(hello(i,"am"));
}
```

- a) iam
- b) i"am"
- c) am
- d) "am"

Answer: b



360. What will be the output of the following C code?

```
#define F abc
#define B def
#define FB(arg) #arg
#define FB1(arg) FB(arg)
main()
{
    printf(FB(F B));
    FB1(F B);
}
```

- a) F B
- b) Error
- c) FB
- d) "FB"

Answer: a

361. What will be the output of the following C code?

```
#define display(text) "$" #text
main()
{
    printf(display(hello world));
}
```

- a) hello world
- b) \$helloworld
- c) \$hello world
- d) error

Answer: c

362. What will be the output of the following C code?

```
#include<stdio.h>
#define max 100
main()
{
    #ifdef max
    printf("hello");
}
```

- a) 100
- b) hello
- c) "hello"
- d) error

Answer: d

363. \_\_\_\_\_ is the preprocessor directive which is used to end the scope of #ifdef.

- a) #elif

- b) #ifndef
- c) #endif
- d) #if

Answer: c

364. What will be the output of the following C code?

```
#include<stdio.h>
void main()
{
    #ifndef max
    printf("hello");
    #endif
    printf("hi");
}
```

- a) hello
- b) hellohi
- c) error
- d) hi

Answer: b



365. What will be the output of the following C code?

```
#define san 557
main()
{
    #ifndef san
    printf("yes");
    #endif
    printf("no");
}
```

- a) error
- b) yes
- c) no
- d) yesno

Answer: c

366. The preprocessor directive which checks whether a constant expression results in a zero or non-zero value \_\_\_\_\_

- a) #if
- b) #ifdef
- c) #undef
- d) #ifndef

Answer: a

367. What will be the output of the following C code?

```
#include<stdio.h>
#define max 100
void main()
{
    #if(max%10)
    printf("san");
    #endif
    printf("foundry");
}
```

- a) error
- b) san
- c) foundry
- d) sanfoundry

Answer: d

368. The preprocessor directive which is used to remove the definition of an identifier which was previously defined with #define?

- a) #ifdef
- b) #undef
- c) #ifndef
- d) #def

Answer: b

369. What will be the output of the following C code?

```
#define hello 10
void main()
{
    printf("%d",hello);
    #undef hello
    printf("%d",hello);
}
```

- a) 10
- b) hello
- c) error
- d) 1010

Answer: c

370. What will be the output of the following C code?

```
#include <stdio.h>
#define a 2
main()
{
    int r;
    #define a 5
    r=a*2;
    printf("%d",r);
}
```

- a) 10
- b) 4
- c) 2
- d) 5

Answer: a

371. What will be the output of the following C code if the value of 'p' is 10 and that of 'q' is 15?

```
#include<stdio.h>
int main()
{
    int p,q;
    printf("Enter two numbers\n");
    scanf("%d",&p);
    scanf("%d",&q);
    #if(4<2)
    printf("%d",p);
    #elif(2>-1)
    printf("%d",q);
    #else
    printf("bye");
    #endif
}
```

- a) 10
- b) 15
- c) bye
- d) error

Answer: b

372. What will be the output of the following C code?

```
#include<stdio.h>
#define san 10
main()
{
    #ifdef san
    #define san 20
    #endif
    printf("%d",san);
}
```

- a) 10
- b) 20
- c) Error
- d) 1020

Answer: b.

373. What will be the output of the following C code?

```
#include<stdio.h>
#define hello
main()
{
    #ifdef hello
    #define hi 4
    #else
    #define hi 5
    #endif
    printf("%d",hi);
}
```

- a) 4
- b) 5
- c) 45
- d) error

Answer: a

374. The purpose of the preprocessor directive #error is that \_\_\_\_\_

- a) It rectifies any error present in the code
- b) It rectifies only the first error which occurs in the code
- c) It causes the preprocessor to report a fatal error
- d) It causes the preprocessor to ignore an error

Answer: c

375. What will be the output of the following C code?

```
#include<stdio.h>
#define max 20
main()
{
    #ifndef max
    #define min 10
    #else
    #define min 30
    #endif
    printf("%d",min);
}
```

- a) 10
- b) 20
- c) 30
- d) error

Answer: c

376. What will be the output of the following C code?

```
#include<stdio.h>
#define hello 10
main()
{
    #ifdef hello
    #undef hello
    #define hello 100
    #else
    #define hello 200
    #endif
    printf("%d",hello);
}
```

- a) Error
- b) 10
- c) 100
- d) 200

Answer: c

377. What will be the output of the following C code?

```
#include<stdio.h>
#define sf 10
main()
{
    if(sf==100)
        printf("good");
    else
    {
        printf("bad");
        sf=100;
    }
    printf("%d",sf);
}
```

- a) 100
- b) bad
- c) 10
- d) error

Answer: d

378. What will be the output of the following C code?

```
#include<stdio.h>
void f()
{
    #define sf 100
    printf("%d",sf);
}
int main()
{
    #define sf 99;
    f();
    printf("%d",sf);
}
```

- a) error
- b) 100
- c) 99
- d) 10099

Answer: a

379. What will be the output of the following C code?

```
#include<stdio.h>
#define INDIA 1
#define US 2
#define CC US
main()
{
    #if CC==INDIA
    printf("Rupee");
    #elif CC==US
    printf("Dollar");
    #else
    printf("Euro");
    #endif
}
```

- a) Euro
- b) Rupee
- c) Dollar
- d) Error

Answer: c

380. What will be the output of the following C code?

```
#define sqr(x) x*x
main()
{
    int a1;
    a1=25/sqr(5);
    printf("%d",a1);
}
```

- a) 25
- b) 1
- c) 5
- d) error

Answer: a

381. Which of the following is not a preprocessor directive?

- a) #error
- b) #pragma
- c) #if
- d) #ifelse

Answer: d



<b>Prepared By</b>	<b>Verified By</b> Module Coordinator	<b>Re-Verified By</b> Academic Coordinator	<b>Approved By</b> Mr. Tamboli S.B. HoD CO

