MADE EASY PUBLICATIONS

POSTAL BOOK PACKAGE 2026

CONTENTS

COMPUTER SCIENCE & IT

Objective Practice Sets

Programming and Data Structures

Note: This book contains copyright subject matter to MADE EASY Publications, New Delhi. No part of this book may be reproduced, stored in a retrieval system or transmitted in any form or by any means. Violators are liable to be legally prosecuted.

1 CHAPTER
Programming Methodology

Multiple Choice Questions & NAT Questions

1. Consider the following function declaration

int* *f*(int *);

Which of the following is correct about the declaration?

(a) *f* is a function which takes integer pointer as argument and returns integer.
(b) *f* is a function which takes integer pointer as an argument and returns address of an integer.
(c) *f* is a pointer to a function which takes integer pointer as an argument and returns integer.
(d) *f* is a pointer to a function which takes integer pointer as an argument and returns address of an integer.

2. Find the output of the following program:

```
main( )
{
    extern int i;
    i = 20;
pirntf("%d", i);
}
```

(a) Linker error (b) 20
(c) Compiler error (d) None of these

3. Consider the following code?

```
void main( )
{   static int i = 5;
    if (– – i)
    {
        main ( );
        printf ("%d", i);
    }
}
```

How many zero's are printed in the output?

4. Which of the following is correct output for the program code given below?

```
main( )
{
    void pr( );
    pr ( );
    pr ( );
    pr ( );
}
void pr ( )
{
    static int i = 1;
    printf ("%c", (65+ i ++));
}
```

(a) 66, 67, 68 (b) 66, 66, 66
(c) 67, 68, 69 (d) None of these

5. Which of the following are equivalent to the statement?

int *k* = (*i* << 3) + (*j* >> 2)

(a) int *k* = *i* * 8 + *j*/4;
(b) int *k* = *i* * 3 + *j* * 2;
(c) int *k* = *i* * 3 + *j*/2;
(d) int *k* = *i*/8 + *j* * 4;

6. Consider the following foo function and identify the return value of foo function.

```
int foo (unsigned int n)
{   int c, x = 0;
    while (n! = 0)
    {   if (n & 1) x ++;
        n >>= 1;
    }
    return c;
}
```

(a) It counts the total number of bits set in an unsigned integer.
(b) It counts the number of bits which are zero.
(c) It counts the number of occurrences of 01.
(d) It returns the same value as '*n*'.

7. Consider the following code:

```
int f(int a, int b)
{
    if (b == 0) return 1;
    else if (b % 2 == 0)
    {
        return (f(a, b/2) * f(a, b/2));
    }
    else
    {
        return (a * f(a, b/2) * f(a, b/2));
    }
}
```

The return value of $f(2, 10)$ is _____.

8. What is output of the following program?

```
# include <stdio.h>
# define R 10
# define C 20
int main( )
{    int (*P) [R] [C];
     printf("%d", size of (*P));
     getchar( );
     return 0;
}
```

(a) 4 (b) 8
(c) 2 (d) None of these

9. Match **List-I** with **List-II**:

List-I

A. typedef int (* ptr) ( ); ptr *p*;
B. int (* *P*) [4];
C. int * *P* [4];

List-II

1. Pointer to an array of integer
2. Pointer to a function returning an integer
3. Array of pointers, pointing to integer

Codes:

	A	B	C
(a)	1	2	3
(b)	2	1	3
(c)	2	3	1
(d)	1	3	2

10. Consider the following pseudocode program:

```
int i
main ( )
{
    i = 3
    S ( )
    R ( )
}
void S ( )
{
    print i // prints the value of i on the current line of output
    print " " // prints a blank space on the current line of output
}
void R ( )
{
    int i
    i = 2
    S ( )
}
```

What is the output of the program if the pseudocode uses either static (lexical) scoping or dynamic scoping?

	Static Scoping	Dynamic Scoping
(a)	3 2	3 2
(b)	3 3	2 2
(c)	3 3	2 3
(d)	3 3	3 2

11. Consider the following code:

```
int a = 32, b = 2, c = 3;
Switch (X)
{
    Case 2: printf("%d", a);
    Case 4: printf("%d", b);
    Case 6: break;
    Case 8: printf("%d", c);
    default: printf("%d", b);
}
```

Find the missing statement *X*, if the above '*C*' code prints the output as 32.

(a) $b * c$ (b) $b * c - 2$
(c) $b + c * 2$ (d) None of these

12. Which of the following statement is false about 'return' statement?

(a) It terminates the execution of a function.

(b) Control moves back to the calling environment after the return statement execution.

(c) It cannot contain an expression.

(d) It may appear more than once in the same function.

13. Consider the following pseudocode:

```
int i = 0;
main( )
{
    i = 3;
    A( );
    B( );
}
A( ) { print "i"; }
B( ) { int i = 2; A( )}
```

What is the output of the above code if it uses static scoping?

(a) 2, 3 (b) 3, 2

(c) 2, 2 (d) 3, 3

14. Which of the following is a valid switch statement?

(a)
```
switch (i)          // i is an integer
{
    case 1: break;
    case j: break; // j is a variable
}
```

(b)
```
switch (i)          // i is a string
{
    case "abc" : break;
    case "xyz" : break;
}
```

(c)
```
switch (i)          // i is an integer
{
    case 1 : break;
    case 2*4 : break;
}
```

(d) Both (a) and (c)

15. Consider the following code:

```
int main( )
{
    char A[ ] = "gate";
    int x;
    for (x = 0; A[x]; x++)
    {
        printf("%c", A[x]);
    }
}
```

What is the output printed by the code?

(a) gate (b) g

(c) run time error (d) compile time error

16. What will be the output of the following program?

```
#include <stdio.h>
#include <string.h>
int main( )
{
    int X = size of ("MADEEASY");
    int Y = strlen ("MADEEASY");
    printf("%d%d", X, Y)
    return 0;
}
```

(a) 88 (b) 99

(c) 89 (d) 98

17. Consider the following C program:

```
# include <stdio.h>
void f(int x, int * p)
{
    *p = x;
    x = 10;
}
int main( )
{
    int a = 5,  b = 6;
    int  *p = &a, **q;
    *p = 20; q = &p;
    f(a, &b);
    *q = &b;
    *p = 30;
    printf("%d, %d", a, b);
}
```

What is the output product by above C program?

(a) 10, 20 (b) 20, 30

(c) 30, 10 (d) 20, 20

Answers Programming Methodology

1.	(b)	**2.**	(a)	**3.**	(4)	**4.**	(d)	**5.**	(a)	**6.**	(a)	**7.**	(1024)	**8.**	(d)	**9.**	(b)		
10.	(d)	**11.**	(c)	**12.**	(c)	**13.**	(d)	**14.**	(c)	**15.**	(9)	**16.**	(d)	**17.**	(b)	**18.**	(a)		
19.	(d)	**20.**	(21)	**21.**	(d)	**22.**	(d)	**23.**	(240)	**24.**	(9)	**25.**	(5)	**26.**	(c)	**27.**	(c)		
28.	(d)	**29.**	(a)	**30.**	(a)	**31.**	(b)	**32.**	(a)	**33.**	(114)	**34.**	(c)	**35.**	(50)	**36.**	(10)		
37.	(a)	**38.**	(b)	**39.**	(50)	**40.**	(115)	**41.**	(43211234)	**42.**	(c)	**43.**	(d)	**44.**	(b)				
45.	(a)	**46.**	(c)	**47.**	(a)	**48.**	(b)	**49.**	(b)	**50.**	(c)	**51.**	(c)	**52.**	(c)	**53.**	(b)		
54.	(d)	**55.**	(c)	**56.**	(d)	**57.**	(65)	**58.**	(13)	**59.**	(a)	**60.**	(40)	**61.**	(b)	**62.**	(a)		
63.	(a)	**64.**	(c)	**65.**	(166)	**66.**	(c)	**67.**	(c)	**68.**	(290)	**69.**	(1365)	**70.**	(10)	**71.**	(d)		
72.	(51)	**73.**	(23)	**74.**	(c)	**75.**	(61)	**76.**	(d)	**77.**	(b)	**78.**	(15)	**79.**	(0)	**80.**	(300)		
81.	(c)	**82.**	(50)	**83.**	(c)	**84.**	(556)	**85.**	(a)	**86.**	(b)	**87.**	(a)	**88.**	(302011)				
89.	(d)	**90.**	(106)	**91.**	(d)	**92.**	(17)	**93.**	(b)	**94.**	(b)	**95.**	(d)	**96.**	(b)	**97.**	(d)		
98.	(a, b, c)	**99.**	(a, c)	**100.**	(b)														

Explanations Programming Methodology

1. (b)

The correct declaration for (a) is int *f*(int ∗)

The correct declaration for (b) is int* *f*(int *);

The correct declaration for (c) is int (∗*f*) (int ∗)

The correct declaration for (d) is int ∗(∗*f*) (int ∗)

2. (a)

Linker error: Undefined symbol-*i*

Extern int *i*; Specifies to the compiler that the memory for *i* is allocated in some other program and that address will be given to the current program at the time of linking. But linker finds that no other variable of name '*i*' is available in any other program with memory space allocated for it. Hence linker error occurred.

3. (4)

The variable '*i*' is declared as static, hence memory for '*i*' will be allocated for only once, as it encounters the statement. The function main( ) will be called recursively unless *i* becomes equal to zero and since main ( ) is recursively called, so the value of static *i*, i.e. 0 will be printed every time the control is returned.

So total 4 times zero is printed.

4. (d)

The correct output is "*BCD*" when the function pr ( ) is first called the value of *i* is initialized to 1. After the pr ( ) completes its execution *i* = 2 is retained for it's next call as "*i*" is static variable.

∴ 65 + 1 = 66 (B)

65 + 2 = 67(C)

65 + 3 = 68(D)

∴ *BCD* is the correct output.

5. (a)

<< and >> are bit wise operators used to multiply and divide by power of 2 respectively (shift operators)

∴ $i << 3 \Rightarrow i * 8$

$j >> 2 \Rightarrow j / 4$

6. (a)

It counts the number of bits set in an unsigned integer.

```
while (n! = 0)
{
    if (n & 1) x ++;
    /* performs bit wise AND operator and if
    condition is satisfied if result
    contains atleast one 1.
```

$n >> = 1$

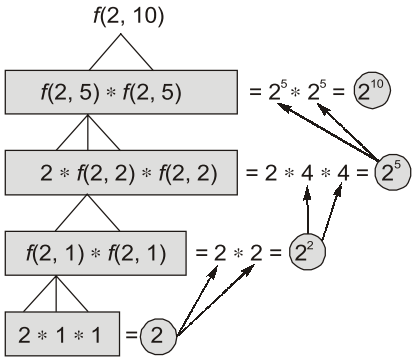
}

x ++; Maintains the count for number of 1's.

$n >>= 1$ Shift the 'n' bit number by 1 bit to right.

7. (1024)

$f(2, 10)$ returns 2^{10} value = 1024



8. (d)

int (*p) [R] [C] ⇒ pointer to an array of array of integer.

Output: 10 * 20 * size of (int) which is 800 for compliers with integer size as 4 bytes and 400 for compilers with integer size as 2 bytes.

The pointer p is de-referenced, hence it yields type of the object. In the present case, it is an array of array of integers.

So, it prints R * C * size of (int).

9. (b)

A : return type is int. It is a pointer to a function.

B : (* P) declares pointer. (* P) [4] is array pointed by pointer.

C : * P [4] declares array of pointers.

10. (d)

Using static scoping: First print prints the global i whose value is 3. Second print prints the global i whose value is 3.

Using dynamic scoping: First print prints the global i whose value is 3. Second print prints the local i whose value is 2 (from the function it was called).

11. (c)

$X : b + c * 2$ is 8

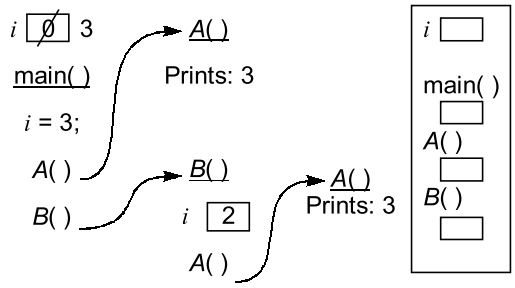
Case 8: prints 3 then default case prints 2

∴ Output prints 32.

12. (c)

Return statement can contain an expression.

13. (d)



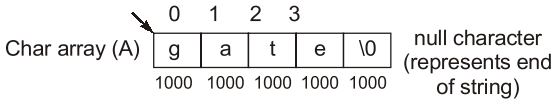
Output printed by the code: 3, 3

14. (c)

Only constants or enums can be used with cases of switch. 2 * 4 is a constant expression.

15. (a)

Let string gate be stored from memory location 1000.



Given loop prints string from A[0] to A[3], i.e., "gate"

16. (d)

Size of ( ) returns length of string including null character (\°). While strlen ( ) returns length of string without including null character.

So here output is $X = 9$, $Y = 8$.

17. (b)

After Execution

main ( )	a	b	p	q
int $a = 5$, $b = 6$;	5	6		
int *p = &a, **q;	5	6	&a	–
*p = 20; q = &p;	20	6	&a	&p
f(a, &b);	20	20	&a	&p
*q = &b;	20	20	&b	&p
*p = 30;	20	30	&b	&p

$a = 20$, $b = 30$