1. **What will be the output of the following pseudocode?**

```
1. Integer a, b, c, d
2. Set a = 10, b = 20, c = 30, d = 40
3. a = b * a
4. b = d - c
5. c = b * 2
6. a = a ^ c
7. b = b - 2
8. b = b << 1
9. c = (c & a) + (a << 1)
10. if(c > 5 || b < 10)
11.         d = a + b + c - 5
12. end if
13. d = d + a
14. Print d
```

[Note- &: bitwise AND - The bitwise AND operator (&) compares each bit of the first operand to the corresponding bit of the second operand. If both bits are 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
<< is left shift operator, it takes two numbers, left shifts the bits of the first operand, the second operand decides the number of places to shift.
||: Logical OR - The logical OR operator (||) returns the Boolean value TRUE (or 1) if either or both operands are true and return FALSE (or 0) otherwise.
^ is the bitwise exclusive OR operator that compares each bit of its first operand to the corresponding bits of its second operand. If one bit is 0 and the other bit is 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.]

- ○ 927
- ○ 665
- ◉ 911
- ○ 129

2. **What will be the output of the following pseudocode?**

```
1. Integer a, b, c
2. Set a = 2, b = 40, c = 0
3. b = c + 2
4. if(a)
5.         c = 1
6. End if
7. Print a - b + c
```

[Note: If(x) gets executed if the value inside if(), i.e., x is not zero.]

- ○ 4
- ◉ 1
- ○ -2
- ○ 11

3. What will be the output of the following pseudocode?

```
1. Integer a, b, c
2. Set a = 4, b = 1, c = 2
3. if(b ^ (c & a) && a ^ (c & b))
4.         c = a + a
5.         a = c + c
6. Else
7.         c = b + b
8.         b = c + c
9. End if
10.  Print a + b + c
```

[Note-&&: Logical AND - The logical AND operator (&&) returns the Boolean value true (or 1) if both operands are true and return false (or 0) otherwise.
&: bitwise AND - The bitwise AND operator (&) compares each bit of the first operand to the corresponding bit of the second operand. If both bits are 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
^ is the bitwise exclusive OR operator that compares each bit of its first operand to the corresponding bits of its second operand. If one bit is 0 and the other bit is 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
If(x) gets executed if the value inside if(), i.e., x is not zero.]

- ○ 22
- ○ 34
- ○ 31
- ◉ 25

4. What will be the output of the following pseudocode?

```
1. Integer a, b, c
2. Set a = 10, b = 1, c = 2
3. if(b&c && a&b && a<<1)
4.         c = c ^ a
5.         a = 0
6. Else
7.         c = 0
8.         a = 2
9. End if
10.  Print a + b + c
```

[Note-&&: Logical AND - The logical AND operator (&&) returns the Boolean value true (or 1) if both operands are true and return false (or 0) otherwise.
<< is left shift operator, it takes two numbers, left shifts the bits of the first operand, the second operand decides the number of places to shift.
&: bitwise AND - The bitwise AND operator (&) compares each bit of the first operand to the corresponding bit of the second operand. If both bits are 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
^ is the bitwise exclusive OR operator that compares each bit of its first operand to the corresponding bits of its second operand. If one bit is 0 and the other bit is 1, the corresponding

result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
If(x) gets executed if the value inside if(), i.e., x is not zero.]

- (●) 3
- ( ) 21
- ( ) 1
- ( ) 11

5. What will be the output of the following pseudocode?

```
1. Integer a, b, c
2. Set a = 1, b = 4, c = 2
3. if(1 && 1)
4.         c = (a & b) + (a ^ b)
5.         if(c)
6.                 c = a
7.         End if
8. End if
9. Print c + a + b
```

[Note-&&: Logical AND - The logical AND operator (&&) returns the Boolean value true (or 1) if both operands are true and return false (or 0) otherwise.
&: bitwise AND - The bitwise AND operator (&) compares each bit of the first operand to the corresponding bit of the second operand. If both bits are 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
^ is the bitwise exclusive OR operator that compares each bit of its first operand to the corresponding bits of its second operand. If one bit is 0 and the other bit is 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
If(x) gets executed if the value inside if(), i.e., x is not zero.]

- (●) 6
- ( ) 7
- ( ) 8
- ( ) 5

6. What will be the output of the following pseudocode?

```
1. Integer a, b
2. Set a = 3, b = 3
3. a = b
4. b = a
5. if(2 ^ 1 ^ 3)
6.         a = a +1
7. Else
8.         b = b - 1
9. End if
10.  Print a + b
```

[Note- ^ is the bitwise exclusive OR operator that compares each bit of its first operand to the corresponding bits of its second operand. If one bit is 0 and the other bit is 1, the corresponding

result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
If(x) gets executed if the value inside if(), i.e., x is not zero.]

- ○ 7
- ◉ 5
- ○ 6
- ○ 4

7. What will be the output of the following pseudocode?

```
1. Integer a, b , c
2. Set a = 1, b = 2, c = 5
3. if(a mod 1 && a^1)
4.         b = b - c
5. End if
6. if(a mod 1 || 1&a )
7.         c = c + a
8. End if
9. Print a + b + c
```

[Note-&&: Logical AND - The logical AND operator (&&) returns the Boolean value true (or 1) if both operands are true and return false (or 0) otherwise.
||: Logical OR - The logical OR operator (||) returns the Boolean value TRUE (or 1) if either or both operands are true and return FALSE (or 0) otherwise.
mod finds the remainder after the division of one number by another. for example, the "5 mod 2" would evaluate to 1 because 5 divided by 2 leaves a quotient of 2 and a remainder of 1.
&: bitwise AND - The bitwise AND operator (&) compares each bit of the first operand to the corresponding bit of the second operand. If both bits are 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
^ is the bitwise exclusive OR operator that compares each bit of its first operand to the corresponding bits of its second operand. If one bit is 0 and the other bit is 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
If(x) gets executed if the value inside if(), i.e., x is not zero.]

- ○ 19
- ◉ 9
- ○ 13
- ○ 8

8. What will be the output of the following pseudocode?

```
1. Integer a, b, c
2. Set a = 2, b = 4, c = 2
3. b = a + 1
4. a = c + 1
5. c = b + 1
6. if(a + 2)
7.         if(b + 2)
8.                 a = b + 2
9.         End if
```

```
10.         b = c + 2
11.         if(c + 5)
12.                 a = b + 2
13.         End if
14. End if
15. Print a + b + c
```

[Note: If(x) gets executed if the value inside if(), i.e., x is not zero.]

- (●) 18
- ( ) 13
- ( ) 22
- ( ) 26

9. What will be the output of the following pseudocode?

```
1. Integer a, b
2. Set a=1, b =2
3. if(b+11>a || a-11 || 0 || 1)
4.         b = b+a
5. Else
6.         b = a-b
7. End if
8. Print b-a
```

[Note: ||: Logical OR- The logical OR operator (11) returns the Boolean value TRUE (or 1) if either or both operands is TRUE and returns FALSE (or 0) otherwise
If(x) gets executed if the value inside if(), i.e., x is not zero]

- ( ) 7
- (●) 2
- ( ) -15
- ( ) 17

10. What will be the output of the following pseudocode?

```
1. Integer a, b, c
2. Set a = 3, b = 1, c=3
3. if (a & b & c)
4.         a= a & b & c
5. End if
6. if (a ^ b ^ c)
7.         a= a ^ b ^ c
8. End if
9. Print a - b + c
```

[Note- & bitwise AND-The bitwise AND operator (&) compares each bit of the first operand to the corresponding bit of the second operand. If both bits are 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
^ is the bitwise exclusive OR operator that compares each bit of its first operand to the corresponding bit of its second operand. If one bit is 0 and the other bit is 1, the corresponding

result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
If(x) gets executed if the value inside if(), i.e., x is not zero]

- ○ 9
- ○ 14
- ◉ 5
- ○ 1

11. What will be the output of the following algorithm?

```
Start
Declare a, I and b
for I =0 to 4
   Increment a by 1
   if I = 3 then
      print hello
      get out of the loop
   End if
End for
print a
```

- ○ 2
- ○ 4
- ◉ hello4
- ○ hello

12. What will be the out put of following pseudocode ?

```
integer a,b,c
set b=10
for(each a from 1 to 4)
   b=b+a
end for
c=b/5
print c
```

- ○ 20
- ○ 5
- ◉ 4
- ○ 10

13. What will be the out put of following pseudocode ?

```
char[]text='TESTSTRING'
integer a,c
char ch='T'
c=0
for(each a from 0 to length of text)
   if (text[a]==ch)
      c=c+1
```

```
   end if
end for
if (c>0)
   print(c)
else
   print"0"
```

- ○ 6
- ○ 1
- ◉ 3
- ○ 10

14. What will be the output of the following pseudocode?

```
Integer a, b, c, d, e=0
Set a=50 , b=3, c=3
while(c>0)
   d=a mod b
   e= e+d + a
   c= c - 1
End while
Print e
```

- ○ 100
- ○ 153
- ◉ 156
- ○ 52

15. What is the output of the following pseudo-code ?

```
input a[ ]={12,14,16,18} and set sum =0
for i=0 to n
   if( a[i] mod 2 equals 0)
   sum=sum+a[i]
end for loop
print sum
```

- ○ 1
- ◉ 60
- ○ 0
- ○ 18

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

```
Integer a,b,c
Set a=6,b=84
while(b>0)
   b=b/2
   a=a+6
   c=a+b
```

```
    while(c>40)
      if(c mod 2 IS EQUAL TO 0)
        Print a
      else
        Print b
    c=c/10
    End while
End while
Print c
```

- ○ 12 4
- ◉ 12 1 48 4
- ○ 12 1 4
- ○ 48 4

17. What will be the output of the following pseudocode?

```
Integer a,b
Set a=2, b=50
  while(b>0)
    a = b MOD 2 +a
    if( a MOD 3 IS EQUAL TO 0)
      Print (a)
    else
      Print(b-1)
    b=b/5
    a=a+1
end while
```

- ◉ 49, 3, 1
- ○ 3, 3, 3
- ○ 50, 10, 2
- ○ 50, 3, 2

18. What will be the output of the following pseudocode?

```
int m = 9, n = 6
m = m + 1
n = n - 1
m = m + n
if(m > n)
  print m
else
  print n
```

- ○ 5
- ◉ 15
- ○ 16
- ○ 4