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 pseudocode?

```
1. Integer a, b, v, c
2. Set a = 9, v = 27
3. while(v > 5)
4.         a = a + v
5.         c = a - 10
6.         while(c > 7)
7.                 b = v + c
8.                 c = c - 60
9.         end while
10.        v = v/3
11. end while
12. Print a, c, v
```

- ◉ 45 -25 3
- ○ 45 25 3
- ○ None of the mentioned options
- ○ 89 -41 4

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

```
1. Integer p, q, r
2. Set q = 13
3. for(each p from 1 to 4)
4.         r = q mod p
5.         p = p + 5
6.         q = p + r
7. end for
8. r = q / 5
9. Print q, r
```

[Note: 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]

- ◉ 6 1
- ○ 6 4
- ○ 7 2
- ○ 1 3

13.

What will be the output of the following pseudocode?

```
1. Integer a, b, c
2. Set a = 1, b = 1
3. for(each c from 3 to 6)
4.         a = a + b
5.         if(a<0 || b>0)
6.                 b = 10
7.                 a = 11
8.                 Continue
9.         End if
10.        b = a
11.        a = b
12. End for
13. Print a + b
```

[Note- Continue: When a continue statement is encountered inside a loop, control jumps to the beginning of the loop for next iteration, skipping the execution of statements inside the body of the loop for the current iteration.
||: 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.]

- (●) **21**
- ( ) **22**
- ( ) **14**
- ( ) **34**

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

```
1. Integer a, b, c
2. Set a = 2, b = 3
3. for(each c from 3 to 5)
4.         if(c > 3 || b > 3)
5.                 a = a + c
6.         End if
7.         b = b - 1
8.         b = b + a
9. End for
10. b = b + 1
11. Print a + b
```

[Note- ||: 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.]

- ( ) **33**
- (●) **31**
- ( ) **37**
- ( ) **30**

15.

What will be the output of the following pseudocode?

```
1. Integer a, b, c
2. Set a = 1, b = 2
3. for(each c from -1 to 1)
4.         a = a + c
5.         if(a < 1 && b < a)
6.                 Continue
7.         Else
8.                 a = a + 1
9.         End if
10.                a = a + c
11. End for
12. Print 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.
Continue: When a continue statement is encountered inside a loop, control jumps to the beginning of the loop for next iteration, skipping the execution of statements inside the body of the loop for the current iteration.]

- ○ 7
- ○ 18
- ○ 3
- ◉ 6

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

```
1. Integer a, b, c
2. Set a = 1, b = 2
3. for(each c from 1 to 3)
4.         if(a + (b ^ c))
5.                 a = a + 1
6.                 if(c ^ 2)
7.                         Continue
8.                 End if
9.         End if
10.        a = a + 1
11. End for
12. a = a + 1
13. Print a + b
```

[Note- Continue: When a continue statement is encountered inside a loop, control jumps to the beginning of the loop for next iteration, skipping the execution of statements inside the body of the loop for the current iteration.
^ 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.]

- ○ 12

- ○ 27
- ○ -6
- ◉ 8

17. How many times A will be printed for n = 5?

```
1. def fun1(int n)
2.         Integer i
3.         Set i = 0
4.         if(n greater than 1)
5.                 fun1(n - 1)
6.         end if
7.         for(each i from 0 to n-1)
8.                 Print “A”
9.         end for
10.  end function fun1()
```

- ○ 17
- ○ 13
- ◉ 15
- ○ 14

18. What will be the output of the following pseudocode for a = 2, b = 9?

```
1. Integer funn(Integer a, Integer b)
2.         if(b > 3 && 1)
3.                 return funn(a - 1, b - 7)
4.         Else
5.                 return (a & b) + (a ^ b)
6.         End if
7. End function funn()
```

[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
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 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.]

- ○ 14
- ◉ 3
- ○ -2
- ○ 4