Part A

Q1.1. **Mark 1**

Which of the following are not valid variable name(s) in C language?

_ , _name, 100%valid, while_for, roll-number, main, invalid

100%valid, roll-number

Q1.2. **Mark 1**

Write the output of the following statement.

```
printf ("Values = %d %d %f %f ", 8/3*3, 2+5/2%3-1, (float) (7/2), (float)7/2 );
```

Values = 6 3 3.00 3.50

Q1.3. **Mark 1**

Write the output of the following code statement.

```
printf ("Values = %d %o %x ", 527, 527, 527);
```

527 1017 20f

Q1.4. **Mark 1**

Write the output of the following code statement.

```
int arr[4] = {10,20,30,40};
int *iptr = &arr[1];
printf("%d %d %d %d", sizeof(arr), sizeof(*iptr), *(iptr+2), arr[2] - *iptr);
```

16 4 40 10

Q1.5. **Mark 1**

Write down the function prototype for which (you can choose any name for the function)

- The return type is a double pointer
- The parameters are as follows (in order): an array of integer variables, a floating-point value, a string and an address of some integer variable

double* func (int arr[], float f, char* str, int* addr);

Q1.6. **Mark 1**

The declaration statement for an ***array of character pointer variables with size 5*** is written as: char* arr_ptr[5];
Calculate the value of **sizeof (arr_ptr**) and **sizeof (*arr_ptr)**.

sizeof (arr_ptr) = 40 **sizeof (*arr_ptr**) = 8

Part B

Q2.1. **Mark 3**

Write down the output of the following code snippet (Collatz conjecture, 1937):

```
int y=12, count=0;
while (y != 1) {
            y = y%2 ? 3*y+1 : y/2 ;
            count++;
            printf("%d  ", y);
}
printf("y = %d", count);                          //calculate this output as your answer
```

```
6  3  10  5  16  8  4  2  1 y = 9
```

Q2.2. **Mark 3**

Write down the output of the following code snippet:

```
int x = 10;
if (x = 1)                { printf ("1st if case: %d", x); }
if (--x)                  { printf ("2nd if case: %d", x); }
else if (x == 1)          { printf (“else-if case: %d”, x); }
else                      { printf ("else case: %d", x); }
```

```
1st if case: 1 else case: 0
```

Q2.3. **Mark 3**

Write down the output of the following code snippet:

```
int a[] = { 4, 1, 3, 2, 3 }, i=4, j;
i = --a[i];
j = a[++a[i] ] ;
printf ("%d,%d,%d", a[i], a[--j], sizeof(a));         //calculate this output as your answer
```

```
4, 1, 20
```

Q2.4. **Mark 3**

Write down the output of the function call Func (2,-3):

```
void Func (int n, int m) {
       printf ("\n %d %d", n, m);                  //this line will generate required output(s)
       if (n==0 && m==0) return;
       if (n>0) return Func (m, n-1);
       if (n<0) return Func (m, n+1);
}
```

```
2 3
3 1
1 2
2 0
0 1
```

Q2.5. **Mark 3**

Write down the output of the following code snippet:

```
int x = 3;
switch (x++){
            default: x = 10;
                           break;
            case 3:        x -= 2;
```

```
          case 100:          if( x== 1) { x = 30; } else { x = 40; }
                             break;
          case 40:++x;
}
printf(“%d”, x);                                  //calculate this output as your answer
```

40

Part C

Q3.1. **Mark 4**

Problem: Check if the sum of even numbers in an array is equals to the sum of the odd numbers in an array

Input: An integer array.

Output: Yes or No

Example:

[10, 13, 11, 14] → Yes   [*11*, *25*, *27*] → No

[0, 20, *15*, *5*, 10, *11*, -*1*] → Yes   [-*11*, -20, -30, 0, *11*, -*23*, -*27*] → Yes

Q3.2. **Mark 4**

Problem: Count the number of zero's in a given digit

Input: An integer value X.

Output: An integer

Example:

X = 100 → 2 X = -2000 → 3 X = 124 → 0 X = -10703 → 2

Q3.3. **Mark 4**

Problem: Check if the given input is part of some twin prime.

Definition of twin prime: Two numbers **x** and **y** are called twin primes if both x and y are individually prime numbers and the difference between **x** and **y** is exactly **2**.

Input: An integer value X.

Output: Yes or No.

Example:

X = 11 → Yes X = 17 → Yes X = 12 → No X = 23 → No