Answer Script

Question No. 01

1. Update the following code such that it swaps the contents of two two variables. You can only write code inside the two comments. Do not modify anything else. The printf function is supposed to print “13 and 5”. So basically the value must be swapped but you can not assign 13 directly to a and 5 directly to b. You can not use other numbers as well. But you can declare an extra variable if you need.

```
int a=5, b=13;
//Write code here

//End of code
printf(“%d and %d” , a, b);
```

Answer No. 01

Solution:

```
int a=5, b=13;

int temp = 5; // declared an extra variable
a = 13;
b = temp;

printf(“%d and %d” , a, b); // It will print “13 and 5”
```

Question No. 02

2. You need to take 4 distinct integer as input. Print the largest and smallest among them. Distinct means all of them are separate integers. None of them is equal to any other of them.
   Sample Input:

123 435 34 612

Sample Output:
Largest = 612
Smallest = 34

Answer No. 02

Source Code of the Solution in C:

```
#include<stdio.h>
// Function for finding maximum
int mx (int x, int y) {
   if(x > y) {
      return x;
   } else {
      return y;
   }
 }

// Function for finding minimum
int mn (int x, int y) {
   if(x < y) {
      return x;
   } else {
      return y;
   }
 }

int main() {
 int a, b, c, d;
 scanf("%d%d%d%d", &a,  &b,  &c,  &d);

 int left_mx = mx(a, b);
 int right_mx = mx(c, d);
```

```
 int left_mn = mn(a, b);
 int right_mn = mn(c, d);

 printf("Largest = %d\n", mx(left_mx, right_mx));
 printf("Smallest = %d\n", mn(left_mn, right_mn));

 return 0;
}
```

Question No. 03

3. You will be given a number as input. You will have to find the number of digits in that number. You will have to use loops to solve this problem as the input can have any number of digits. [Hint: Use the modulus operator. You can extract last digit from any number by using %10 operation on the number.]
For example,
Sample Input:
2346167
Sample Output:
7 digits

Since the input had 7 digits so the output is 7. Please use ***long long int*** as data type for this problem.

Answer No. 03

Source Code of the Solution in C:

```
#include<stdio.h>
int main() {
   long long int n;
   scanf("%d", &n);
   int digits = 0;
```

```
  while(n != 0) {
    int rem = n % 10;
    digits++;
    n = n / 10;
  }

  printf("%d\n", digits);
 return 0;
}
```

Question No. 04

4. This time you need to find the sum of the digits of the input. So look at the sample input output.

Sample Input:

2346167

Sample Output:

29

Explanation: Since, the sum of the digits is 2+3+4+6+1+6+7 = 29. So the output is 29.

Answer No. 04

Source Code of the Solution in C:

```
#include<stdio.h>
int main() {
  long long int n;
  scanf("%d", &n);
  int sum = 0;

  while(n != 0) {
    int rem = n % 10;
    sum += rem;
```

```
    n = n / 10;
  }

  printf("%d\n", sum);
 return 0;
}
```

Question No. 05

5. Let us create a new version of weird algorithm. You will be given an integer, **n** as input. If **n** is even, divide it by two. If **n** is odd, then subtract 1 from **n**. Eventually, it will end at 1. So print the whole sequence. For example,

Sample Input:

123

Sample Output:

123, 122, 61, 60, 30, 15, 14, 7, 6, 3, 2, 1

Follow the whole output format(each numbers separated by commas).

Answer No. 05

Source Code of the Solution in C:

```
#include<stdio.h>
int main() {
  int n;
  scanf("%d", &n);

  printf("%d", n);

  while(1) {
    if(n == 1) {
```

```
        break;
    }
    if(n % 2 == 0) {
        n /= 2;
    }
    else {
        n -= 1;
    }
    printf(", %d\n", n);
  }
  return 0;
}
```

Question No. 06

6. Write a C program that will take two integers as input from you. And then it will show us in output whether any one of those two numbers is divisible by the other or not.

Sample Input 1:

Enter the first number: 13
Enter the second number: 39

Sample Output 1:

The second number is divisible by the first number.

Sample Input 2:

Enter the first number: 38
Enter the second number: 13

Sample Output 2:

None of them are divisible by the other.

Answer No. 06

Source Code of the Solution in C:

```c
#include<stdio.h>
int main() {
    int x, y;
    scanf("%d%d", &x, &y);

    if(x % y == 0) {
      printf("The first number is divisible by the second number.\n");
    }

    else if(y % x == 0) {
      printf("The second number is divisible by the first number.\n");
    }

    else {
      printf("None of them are divisible by the other.\n");
    }
   return 0;
}
```

Question No. 07

7. Take two integers as input and print their GCD as the output. We know, GCD of two numbers is the greatest common divisor of two numbers. You can use the logic from Question no. 6 to solve this problem.

   Sample Input 1:

   12 18

   Sample Output 1:

   The GCD of 12 and 18 is 6.

Explanation:
Since 6 is the largest number which divides both 12 and 18 so 6 is output.

Answer No. 07

Source Code of the Solution in C:

```
#include <stdio.h>
int main()
{
 int n1, n2;
 scanf("%d%d", &n1, &n2);

 int gcd;

 for (int i = 2; i <= n1 && i <= n2; i++) {
  if(n1 % i == 0 && n2 % i == 0) {
   gcd = i;
  }
 }

 printf("The GCD of %d and %d is %d\n", n1, n2, gcd);
 return 0;
}
```

Question No. 08

8. Take two integers as input and print their LCM as the output. We know, LCM of two numbers is the least common multiple of two numbers. You can use the logic from Question no. 6 to solve this problem.
Sample Input 1:
12 18
Sample Output 1:

The LCM of 12 and 18 is 36.

Explanation:
Since 36 is the least number which is divisible by both 12 and 18 so 36 is the output.

Answer No. 08

Source Code of the Solution in C:

```
#include <stdio.h>
int main()
{
 int n1, n2;
 scanf("%d%d", &n1, &n2);

 int lcm, mx, mn;

 if (n1 > n2) {
  mx = n1, mn = n2;
 } else {
  mx = n2, mn = n1;
 }

 int c = 1;
 while(1) {
  int temp = mx * c;
  if (temp % mn == 0) {
   lcm = temp;
   break;
  }
  c++;
 }

 printf("The LCM of %d and %d is %d\n", n1, n2, lcm);
 return 0;
```

```
}
```

Question No. 09

9. Write a C program to print all the factors of a number taken as input.

Sample Input 1:
12

Sample Output 1:
The factors of 12 are: 1, 2, 3, 4, 6, 12.

Sample Input 2:
39

Sample Output 2:
The factors of 39 are: 1, 3, 13, 39.

Answer No. 09

Source Code of the Solution in C:

```
#include <stdio.h>
int main()
{
 int n;
 scanf("%d", &n);

 printf("The factors of %d are: ", n);
 for (int i = 1; i < n; i++) {
  if (n % i == 0) {
   printf("%d, ", i);
  }
 }
 printf("%d.\n", n);
```

```
 return 0;
}
```

Question No. 10

10. Primality Testing is one of the most common and the most important problem in the world of Number Theory and also in basic programming. In this problem you will be given an integer as input and you need to find out whether the number is prime or not. Just for your information, prime numbers are the numbers that have only two factors which are 1 and themselves. For example, 37 is a prime number because it has no factors other than 1 and 37. On the other hand 39 is not a prime number called composite number since 39 has factors other than 1 and 39 such as 3 and 13.

Sample Input 1:

12

Sample Output 1:

Composite

Sample Input 2:

1009

Sample Output 2:

Prime

Answer No. 10

Source Code of the Solution in C:

```
#include <stdio.h>
int main()
{
 int n;
 scanf("%d", &n);

 if (n == 1) {
  printf("Composite\n");
  return 0;
```

```
 }

 for (int i = 2; i <= n /2; i++) {
  if (n % i == 0) {
   printf("Composite\n");
   return 0;
  }
 }

 printf("Prime\n");

 return 0;
}
```