WEEK 2 TUTORIAL – BASIC C PROGRAMMING AND CONTROL FLOW

1. **(linearSystem)** Write a C program that computes the solutions for x and y in the linear system of equations:

   $a_1x + b_1y = c_1$
   $a_2x + b_2y = c_2$

   The solutions for x and y are given by:

   $$x = \frac{b_2c_1 - b_1c_2}{a_1b_2 - a_2b_1} \qquad \text{and} \qquad y = \frac{a_1c_2 - a_2c_1}{a_1b_2 - a_2b_1}$$

   The program reads in a_1, b_1, c_1, a_2, b_2 and c_2, and then computes and prints the solutions. In your program, if the denominator (a_1b_2 - a_2b_1) of the above equations is zero, then it prints an error message "Unable to compute because the denominator is zero!".

   A program template is given below.

   ```
   #include <stdio.h>
   #include <math.h>
   int main()
   {
      /* Write your program code here */
      return 0;
   }
   ```

   Sample input and output sessions are given below:

   (1) Test Case 1:
   ```
   Enter the values for a1, b1, c1, a2, b2, c2:
   1 1 1 5 7 9
   x = -1.00 and y = 2.00
   ```

   (2) Test Case 2:
   ```
   Enter the values for a1, b1, c1, a2, b2, c2:
   1 1 2 2 3 3
   x = 3.00 and y = -1.00
   ```

2. (**countChars**) Write a C program that reads in character by character from an input source, until '#' is entered. The output of the program is the number of English letters and the number of digits that appear in the input.

   A sample program is given below:

   ```
   #include <stdio.h>
   int main()
   {
      int ccount = 0, dcount = 0;
      char ch;
      printf("Enter your characters (# to end): \n");

      /* Write your program code here */

      printf("The number of digits: %d\n", dcount);
      printf("The number of letters: %d\n", ccount);
      return 0;
   }
   ```

   Some sample input and output sessions are given below:

(1) Test Case 1:

```
Enter your characters (# to end):
happy 34567 fans#
The number of digits: 5
The number of letters: 9
```

(2) Test Case 2:

```
Enter your characters (# to end):
1a2b3c#
The number of digits: 3
The number of letters: 3
```

3. **(printPattern)** Write a C program that reads a positive number `height` between 1 and 9 as its input value, and prints a triangular pattern according to `height`. Note that only 1, 2 and 3 are used to generate the patterns. For example, when height = 3, it will print the following pattern:

```
1
22
333
```

while height = 7 will print the following pattern:

```
1
22
333
1111
22222
333333
1111111
```

A sample program template is given below.

```
#include <stdio.h>
int main()
{
   int row, col, height;
   int num = 0;
   printf("Enter the height: \n");
   scanf("%d", &height);
   printf("Pattern: \n");

   /* Write your program code here */

   return 0;
}
```

Sample input and output sessions are given below:

(1) Test Case 1:

```
Enter the height:
3
Pattern:
1
22
333
```

(2) Test Case 2:

```
Enter the height:
7
Pattern:
1
```

```
22
333
1111
22222
333333
1111111
```

4. **(computeSeries)** Write a C program that computes the value of e^X according to the following formula:

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots\ldots + \frac{x^{10}}{10!}$$

Sample input and output sessions are given below:

(1) Test Case 1:

```
Enter x:
0.9
Result = 2.46
```

(2) Test Case 2:

```
Enter x:
0
Result = 1.00
```

(3) Test Case 3:

```
Enter x:
-0.9
Result = 0.41
```

A sample program template is given below.

```
#include <stdio.h>
int main()
{
   int n, denominator = 1;
   float x, result = 1.0, numerator = 1.0;

   printf("Enter x: \n");
   scanf("%f", &x);

   /* Write your program code here */

   printf("Result = %.2f\n",  result);
   return 0;
}
```