SVKM's NMIMS

Mukesh Patel School of Technology Management and Engineering, Mumbai

Programming for Problem Solving (Exp 8 - 2)

Roll No: J001	**Name: Adith Ramakrishna**
Program: B. Tech Data Science (1st)	**Batch: J1**
Date of Experiment: 05/11/2022	**Date of Submission: 11/11/2022**

Task 1:

```
#include <iostream>

using namespace std;

bool armstrong(int);

int main() {
   int n;
   cout << "Enter a Number: ";
   cin >> n;
   if (armstrong(n)) {
      cout << "It is an Armstrong Number!";
   } else {
      cout << "It is not an Armstrong Number.";
   }
   return 0;
}

bool armstrong(int num) {
   int temp = num, r, sum = 0;
   while (num > 0) {
      r = num % 10;
      sum = sum + (r * r * r);
      num = num / 10;
   }
   if (temp == sum) {
      return true;
   } else {
      return false;
   }
}
```

Task 2:

```
#include <iostream>

using namespace std;

int factorial(int);

int main() {
   int sum;
   for (int i = 1; i <= 5; i++) {
      sum = factorial(i) / i + sum;
   }
   cout << "Sum: " << sum << endl;
   return 0;
}

int factorial(int n) {
   int ans = 1;
   for (int i = 2; i <= n; i++)
      ans *= i;
   return ans;
}
```

Task 3:

```
#include <iostream>

using namespace std;

int smallest(int arr[], int length);
```

```
int main() {
   int size_arr;
   cout << "Enter number of elements: ";
   cin >> size_arr;
   cout << endl;
   int array[size_arr];
   for (int i = 0; i < size_arr; i++) {
      cout << "\nEnter element no " << i + 1 << ": ";
      cin >> array[i];
   }
   cout << "\nThe Smallest Element is: " << smallest(array, size_arr) <<
endl;
}

int smallest(int arr[], int length) {
   int min = arr[0];
   for (int i = 0; i < length; i++) {
      if (arr[i] < min) {
         min = arr[i];
      }
   }
   return min;
};
```

Homework Questions:

1:

```
#include <iostream>
using namespace std;
```

```
void calc(int[][10], int, int);

int main() {
   int two_dim_arr[10][10], size_arr_x, size_arr_y;
   cout << "Enter the size of the array (x,y): ";
   cin >> size_arr_x >> size_arr_y;
   cout << endl;
   for (int x = 0; x < size_arr_x; x++) {
      for (int y = 0; y < size_arr_y; y++) {
         cout << "Enter the value at (" << x + 1 << ", " << y + 1 << "): ";
         cin >> two_dim_arr[x][y];
      }
   }
   calc(two_dim_arr, size_arr_x, size_arr_y);
}

void calc(int matrix[][10], int row, int col) {
   int sum = 0, count = 0;
   for (int i = 0; i < row; i++)
      for (int j = 0; j < col; j++) {
         if (i < j) {
            sum = sum + matrix[i][j];
            count++;
         }
      }
   cout << "\n\nSum: " << sum;
   cout << "\nAverage: " << sum / count;
}
```

2:

A function declaration tells the compiler about a function's name, return type, and parameters. A function definition provides the actual body of the function. Calling a function is the execution of the defined function body.

Declaration:

```
int sum (int, int);
```

Definition:

```
int sum (int a, int b){
  int c;
  c = a + b;
  return c;
}
```

Calling:

```
int sum = sum(10, 15);
```

3: The different types of functions are:

Functions without arguments and without return values

```
void sum (){
  int a,b,c;
  cout << "Enter 2 numbers: ";
  cin >> a >> b;
  c = a + b;
  cout << c;
}
```

Functions without arguments and with return values

```
int sum (){
  int a,b,c;
  cout << "Enter 2 numbers: ";
```

```
  cin >> a >> b;
  c = a + b;
  return c;
}
```

Functions with arguments and without return values

```
void sum (int a, int b){
  int c;
  c = a + b;
  cout << c;
}
```

Functions with arguments and with return values

```
int sum (int a, int b){
  int c;
  c = a + b;
  return c;
}
```