Assignment 03

3.1 Define a class String that could work as an user-defined string type. Include constructors that will enable us to create uninitialized string

String s1; // string with length 0

and also to initialize an object with a string constant at the time of creation like

String s2("Well done!");

Include a function that adds two strings to make a third string. Note that the statement

s2 = s1;

will be perfectly reasonable expression to copy one string to another. Write a complete program to test your class to see that it does the following tasks:

(a) Creates uninitialized string objects.

(b) Creates objects with string constants.

(c) Concatenates two strings properly.

(d) Displays a desired string object.

```
#include <iostream>
#include <cstring>
using namespace std;
class String {
    char *str;
    int length;

public:
    String() {
        length = 0;
        str = new char[1];
        str[0] = '\0';
    }

    String(const char *s) {
        length = strlen(s);
        str = new char[length + 1];
        strcpy(str, s);
    }
```

```
    String(const String &s) {
        length = s.length;
        str = new char[length + 1];
        strcpy(str, s.str);
    }

    String add(const String &s) {
        String temp;
        temp.length = length + s.length;
        delete[] temp.str;
        temp.str = new char[temp.length + 1];
        strcpy(temp.str, str);
        strcat(temp.str, s.str);
        return temp;
    }

    void display() {
        cout << str << endl;
    }

    ~String() {
        delete[] str;
    }
};

int main() {
    String s1;
    String s2("Well done!");
    String s3(" Keep it up!");
    String s4 = s2.add(s3);
    s4.display();
    return 0;
}
```

3.2 A book shop maintains the inventory of books that are being sold at the shop. The list includes details such as author, title, price and publisher. Whenever a customer wants a book, the salesperson inputs the title and author, and the system searches the list and displays whether it is available in the shop or not. If not, an appropriate message is

displayed. If it is, then the system displays the book details for sale. Design a program using a class called Book with suitable member functions and constructors.

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

class Book {
    string title, author, publisher;
    float price;
public:
    Book(string t, string a, string p, float pr) {
        title = t;
        author = a;
        publisher = p;
        price = pr;
    }

    bool match(string t, string a) {
        return (title == t && author == a);
    }

    void display() {
        cout << "Title: " << title << endl;
        cout << "Author: " << author << endl;
        cout << "Publisher: " << publisher << endl;
        cout << "Price: Rs. " << price << endl;
    }
};

int main() {
    Book books[] = {
        Book("The Alchemist", "Paulo Coelho", "HarperOne", 350),
        Book("C++ Basics", "E. Balaguruswamy", "McGraw Hill", 450)
    };
    string t, a;
    cout << "Enter book title: ";
    getline(cin, t);
    cout << "Enter author name: ";
    getline(cin, a);
```

```
    bool found = false;
    for (int i = 0; i < 2; ++i) {
        if (books[i].match(t, a)) {
            cout << "\nBook Available:\n";
            books[i].display();
            found = true;
            break;
        }
    }

    if (!found)
        cout << "\nBook Not Available.\n";
    return 0;
}
```

3.3 Create a class Complex with data members real and imaginary. Initialize two objects of the Complex class using constructors. Include a static member function addcomplex() to perform the addition of two complex numbers and return the result.

Sample Run:

A = 3.12 +j 5.65

B = 2.75 +j 1.21

C = 5.87 +j 6.86

Here, A, B, and C represent the objects of class Complex.

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

class Complex {
    float real, imag;

public:
    Complex(float r = 0.0, float i = 0.0) {
        real = r;
        imag = i;
    }
```

```
    void display() {
        cout << real << " +j " << imag << endl;
    }

    static Complex addcomplex(const Complex &c1, const Complex &c2) {
        return Complex(c1.real + c2.real, c1.imag + c2.imag);
    }
};

int main() {
    Complex A(3.12, 5.65);
    Complex B(2.75, 1.21);
    Complex C = Complex::addcomplex(A, B);

    cout << "A = "; A.display();
    cout << "B = "; B.display();
    cout << "C = "; C.display();

    return 0;
}
```