Question –

Create a class Complex with the following data members –

i. Real ii. Imaginary

and the following member functions –

i. Input
ii. Output
iii. Sum
iv. Subtract
v. Product

Code –

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

class Complex
{
private:
    double real;
    double imaginary;

public:
    Complex(double r = 0, double i = 0) : real(r), imaginary(i) {}

    void input()
    {
        cout << "Enter real and imaginary parts: ";
        cin >> real >> imaginary;
    }

    void output() const
    {
        cout << real << " + " << imaginary << "i" << endl;
    }

    Complex sum(const Complex &c) const
    {
        return Complex(real + c.real, imaginary + c.imaginary);
    }
```

```
    Complex subtract(const Complex &c) const
    {
        return Complex(real - c.real, imaginary - c.imaginary);
    }

    Complex product(const Complex &c) const
    {
        return Complex(real * c.real - imaginary * c.imaginary, real *
c.imaginary + imaginary * c.real);
    }
};

int main()
{
    Complex c1, c2, result;

    cout << "Enter first complex number:" << endl;
    c1.input();
    cout << "Enter second complex number:" << endl;
    c2.input();

    result = c1.sum(c2);
    cout << "Sum: ";
    result.output();

    result = c1.subtract(c2);
    cout << "Subtraction: ";
    result.output();

    result = c1.product(c2);
    cout << "Product: ";
    result.output();

    return 0;
}
```

Output –

```
Enter first complex number:
Enter real and imaginary parts: 4
5
Enter second complex number:
Enter real and imaginary parts: 4
8
Sum: 8 + 13i
Subtraction: 0 + -3i
Product: -24 + 52i
```

Question –

Implement binary search using –

i. Recursion ii. Without Recursion

Code –

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

int binarySearchRecursive(int arr[], int left, int right, int key) {
    if (left > right)
        return -1; // Element not found

    int mid = left + (right - left) / 2;

    if (arr[mid] == key)
        return mid;
    else if (arr[mid] > key)
        return binarySearchRecursive(arr, left, mid - 1, key);
    else
        return binarySearchRecursive(arr, mid + 1, right, key);
}

int binarySearchIterative(int arr[], int size, int key) {
    int left = 0, right = size - 1;
    while (left <= right) {
        int mid = left + (right - left) / 2;

        if (arr[mid] == key)
            return mid;
        else if (arr[mid] > key)
            right = mid - 1;
        else
            left = mid + 1;
    }
    return -1;
}

int main() {
    int arr[] = {2, 3, 4, 10, 18, 20};
    int size = sizeof(arr) / sizeof(arr[0]);
    int key;
```

```
    cout << "Enter the element to search: ";
    cin >> key;

    int resultRecursive = binarySearchRecursive(arr, 0, size - 1, key);
    int resultIterative = binarySearchIterative(arr, size, key);

    if (resultRecursive != -1)
        cout << "Element found at index (Recursive): " << resultRecursive <<
endl;
    else
        cout << "Element not found (Recursive)" << endl;

    if (resultIterative != -1)
        cout << "Element found at index (Iterative): " << resultIterative <<
endl;
    else
        cout << "Element not found (Iterative)" << endl;

    return 0;
}
```

Output –

```
Enter the element to search: 4
Element found at index (Recursive): 2
Element found at index (Iterative): 2
```