**Logo

Description automatically generated San Francisco Bay University**

**CS360 - Programming in C and C++**

**2022 Summer Midterm Exam**

**Student Name: Student ID:**

1. Create a class called *Money*. It should store money amounts as *long doubles*. Use the

function *mstold()* to convert a money string entered as input into a *long double*, and

the function *ldtoms()* to convert the *long double* to a money string for display. You can call the input and output member functions *getmoney()* and *putmoney()*. Write another member function that adds two *Money* amounts; you can call it *madd()*. Adding *Money* objects is easy: Just add the *long double* member data amounts in two *Money* objects. Write a *main()* program that repeatedly asks the user to enter two money strings, and then displays the sum as a money string. Here’s how the class might look:

*class Money{*

*private:*

*long double mny;*

*public:*

*Money();*

*Money(char s[]);*

*void mstold();*

*void ldtoms();*

*void madd(Money m1, Money m2);*

*void getmoney();*

*void putmoney();*

*};*

1. Define a function template *interpolSearch()* that looks up a given element in a sorted, numeric array. The array elements are of the same type as the template parameter *T*.

The function template has three parameters - the value searched for of type *T*, a pointer to the first array element, and the number of array elements.

The function template returns the index of the first element in the array that corresponds to the searched for value, or *-1* if the value cannot be found in the array.

Implement the function template and verify it in the main function

1. Design & implement, and test a class *Poly* for polynomials. A polynomial has the form

For example:

You’ll need to use an object to store exponents and coefficients. You should implement a constructor that takes a coefficient and an exponent as arguments so that you can write

*Poly c = Poly(3,4) + Poly(2,2) + Poly(7,1) + Poly(-5,0);*

To get the polynomial, you should overload arithmetic operators

*+=, -=* and *+, -* for addition and subtraction. You should overload *\*=* to multiply a polynomial by a constant:

Finally, you should include a member function *at*, that evaluates a polynomial at a specific value for x. For example

*Poly c = Poly(4,2) + Poly(3,1) + Poly(5,0); //*

*double d = c.at(7) //*