PS # 12

Deadline: 21.06.2020 at 23:50

Hand-in Policy:

Create a cpp file for each part of the assignment and save the screenshot while these parts are running.

Your assignment should consist of 3 cpp files and 3 screenshots.

Zip all files of the assignment. Your zip files should look like this

Ps12_studentnumber_name_surname.zip

1)

Write a template-based function that calculates and returns the absolute value of two numeric values passed in. The function should operate with any numeric data types (e.g., float , int , double , char).

2)

The following code uses two arrays, one to store products and another to store product IDs (a better organization would be to use a single array of a class or struct, but that is not the subject of this Programming Project). The function getProductID takes as input the two arrays, the length of the arrays, and a target product to search for. It then loops through the product name array; if a match is

found, it returns the corresponding product ID:

```
int getProductID( int ids[], string names[],int numProducts, string target)
{
        for ( int i=0; i < numProducts; i++)
                {
                        if (names[i] == target)
                                return ids[i];
                }
                return -1; // Not found
        }
        int main() // Sample code to test the getProductID function
        {
                int productIds[] = {4, 5, 8, 10, 13};
                string products[] = {"computer","flash drive",
                "mouse","printer","camera"};
                cout << getProductID(productIds, products, 5, "mouse") << endl;
                cout << getProductID(productIds, products, 5, "camera")
                << endl;
                cout << getProductID(productIds, products, 5, "laptop")
                << endl;
                return 0;
        }
```

One problem with the implementation of the getProductID function is that it returns the special error code of -1 if the target name is not found. The caller might ignore the -1, or later we might actually want to have -1 as a valid product ID number. Rewrite the program so that it throws an appropriate exception when a product is not found instead of returning -1.

3)

A function that returns a special error code is usually better accomplished throwing an exception instead. The following class maintains an account balance.

```
class Account
{
    private:
        double balance;
    public:
        Account()
        {
            balance = 0;
        }
        Account( double initialDeposit)
        {
            balance = initialDeposit;
        }
        double getBalance()
        {
            return balance;
        }
        // returns new balance or -1 if error
        double deposit( double amount)
        {
            if (amount > 0)
                balance += amount;
                else
                        return -1; // Code indicating error
                return balance;
        }
        // returns new balance or -1 if invalid amount
        double withdraw( double amount)
        {
                if ((amount > balance) || (amount < 0))
                        return -1;
                else
                        balance -= amount;
                return balance;
        }
};
```

Rewrite the class so that it throws appropriate exceptions instead of returning -1 as an error code. Write test code that attempts to withdraw and deposit invalid amounts and catches the exceptions that are thrown.