National University of Computer and Emerging Sciences



Lab Exercise 05

For

Object Oriented Programming Lab

|  |  |
| --- | --- |
| Course Instructor(s) | Mr. Usman Ghous |
| Lab Instructor(s) | Mr. Usman Ghous |
| Semester | Spring 2021 |

**FAST School of Computing**

# Instructions:

1. Make a word document with the naming convention “SECTION\_ LAB#\_ROLLNO” and put all your source code and snapshots of its output in it. Make sure your word file is formatted properly.
2. Plagiarism is strictly prohibited.
3. Do not discuss solutions with one another.

# Useful links

|  |
| --- |
| **Question#1** |

A common place to buy candy is from a machine. The machine sells candies, chips, gum, and cookies. You have been asked to write a program for this candy machine.  
The program should do the following:  
1. Show the customer the different products sold by the candy machine.  
2. Let the customer make the selection.  
3. Show the customer the cost of the item selected.  
4. Accept money from the customer.  
5. Release the item.

The machine has two main components: a built-in cash register and several dispensers to hold and release the products.

Define class cashRegister in C++ with the following descriptions:  
**Private Members:**cashOnHand of type integer  
**Public Members:**A default constructor cashRegister() sets the cash in the register to 500.  
A constructor cashRegister(int) sets the cash in the register to a specific amount.  
A function getCurrentBalance() which returns value of cashOnHand  
A function acceptAmount(int) to receive the amount deposited by the customer and update the amount in the register

Define class dispenserType in C++ with the following descriptions :  
**Private Members:**numberOfItems of type integer  
cost of type integer  
**Public Members:**A default constructor dispenserType () sets the cost and number of items in the dispenser to 50 each.  
A constructor dispenserType (int,int) sets the cost and number of items in the dispenser to the values specified by the user.  
A function getNoOfItems() to return the value of numberOfItems.  
A function getCost() to return the value of cost.  
A function makeSale() to reduce the number of items by 1.

When the program executes, it must do the following:  
1. Show the different products sold by the candy machine.  
2. Show how to select a particular product.  
Once the user has made the appropriate selection, the candy machine must act accordingly. If the user has opted to buy a product and that product is available, the candy machine should show the cost of the product and ask the user to deposit the money. If the amount deposited is at least the cost of the item, the candy machine should sell the item and display an appropriate message.  
Divide this program into three functions: showSelection, sellProduct, and main.  
The function sellProduct must have access to the dispenser holding the product (to decrement the number of items in the dispenser by 1 and to show the cost of the item) as well as the cash register (to update the cash). Therefore, this function has two parameters: one corresponding to the dispenser and the other corresponding to the cash register.

|  |
| --- |
| **Question#2** |

Write a class named Employee that has the following member variables:

• name. A string that holds the employee’s name.

• idNumber. An int variable that holds the employee’s ID number.

• department. A string that holds the name of the department where the employee works.

• position. A string that holds the employee’s job title.

* Struct date. Having integer attributes of day, month, year. This will be an object in class.

The class should have the following constructors:

• A default constructor that assigns empty strings (“ ”) to the name, department, and position

member variables, and 0 to the idNumber member variable

* A destructor that displays the message that “I’m a destructor”.

Write appropriate set and get methods to set and retrieve values in these member variables.

Now use these set methods with three different objects to set the class members with the

values given below.

Name ID Number Department Position Date Hired

Adnan Ashraf 47899 Accounting Vice President 19/02/2000

Shahzaib 39119 IT Programmer 29/03/1999

Ziaullah 81774 Manufacturing Engineer 20/02/2005

Display the data for each employee on the console using get methods.

**Question#3**

Design a Class called Jagged that will explore the jagged array concept (2-D dynamic array). Jagged class has attributes int row, column and 1-D array as private data members and following public methods;

* Default Constructor: initialize all members variables with Null values
* Input(), return type will be Jagged and with no input argument. It will ask from user about
* the number of rows and then ask for the number of column of each row. Number of column
* will be different for each row and will be stored in 1-D array. Array size will be equal to the
* number of rows in jagged array.
* Display(), return type should be void and accepts Jagged object. Print the all data of your 2D
* jagged array in a proper and clear way.
* Destructor that will release all the memory.
* Sample Output:

Graphical user interface

Description automatically generated with low confidence

**Question # 4**

1. Some of the characteristics of a book are the title, author(s), publisher,ISBN, price, and year of publication. Design a class bookType that defines the book as an ADT.  
    a) Each object of the class bookType can hold the following information about a book: title, up to four authors,publisher,ISBN,price, and number of copies in stock. To keep track of the number of authors, add another member variable.

b) Include the member functions to perform the various operations on objects of type bookType. For example, the usual operations that can be performed on the title are to show the title, set the title, and check whether a title is the same as the actual title of the book.  
Similarly, the typical operations that can be performed on the number of copies in stock are to show the number of copies in stock, set the number of copies in stock, update the number of copies in stock, and return the number of copies in stock. Add similar operations for the publisher, ISBN, book price, and authors. Add the appropriate constructors and a destructor (if one is needed).

1. Write the definitions of the member functions of the class bookType.
2. Write a program that uses the class bookType and tests various operations on the objects of the class bookType. Declare an array of 100 components of type bookType. Some of the operations that you should perform are to search for a book by its title, search by ISBN, and  
   update the number of copies of a book.

**Question # 5**

In this exercise, you will design a class memberType.

1. Each object of memberType can hold the name of a person, member  
    ID, number of books bought, and amount spent.
2. Include the member functions to perform the various operations on the  
    objects of memberType—for example, modify, set, and show a person’s  
    name. Similarly, update, modify, and show the number of books bought  
    and the amount spent.
3. Add the appropriate constructors.
4. Write the definitions of the member functions of memberType.
5. Write a program to test various operations of your class memberType.

**Question # 6**

Using the classes designed in Programming Exercises 6 and 7, write a  
program to simulate a bookstore. The bookstore has two types of customers:  
those who are members of the bookstore and those who buy books from the  
bookstore only occasionally. Each member has to pay a $10 yearly membership fee and receives a 5% discount on each book purchased.  
For each member, the bookstore keeps track of the number of books  
purchased and the total amount spent. For every eleventh book that a  
member buys, the bookstore takes the average of the total amount of the  
last 10 books purchased, applies this amount as a discount, and then resets the  
total amount spent to 0.  
Write a program that can process up to 1000 book titles and 500 members. Your  
program should contain a menu that gives the user different choices to effectively  
run the program; in other words, your program should be user driven.