|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **National University of Computer and Emerging Sciences, Lahore Campus** | | | | |
| final design | **Course:** | **Computer Programming** | **Course Code:** | **CS-103** |
| **Program:** | **BS (Computer Science)** | **Semester:** | **Spring 2017** |
| **Duration:** | **150 Minutes** | **Total Marks:** | **50** |
| **Paper Date:** | **28 April 2017** | **Weight** | **40%** |
| **Section:** | **C/D** | **Page(s):** | **3** |
| **Exam:** | **Final Term** | **Reg. No** |  |

**Instruction/Notes:**

1. Understanding the question paper is also part of the exam, so do not ask any clarification.
2. Make sure to switch off your mobile phones before the Exam starts.
3. No USB’s are allowed. Please see that the area in your threshold is clean. You will be charged for any material which can be classified as ‘helping in the paper’ found near you.
4. Talking/Discussion is not allowed. It is your responsibility to protect your code and save it from being copied. If you don’t protect it all matching codes are considered copy/cheating cases.
5. You are not allowed to use internet for any purpose.

**Question # 1:**

Write **Linked List base Stack**, use following main program. **(15 points)**

|  |  |
| --- | --- |
| int main()  {  Stack<double> sd;  sd.push(10.1);  sd.push(20.2);  sd.push(30.3);  sd.push(40.4);  Stack<int> si;  si.push(10);  si.push(20);  si.push(30);  si.push(40);  Stack<char> sc;  sc.push('a');  sc.push('b');  sc.push('c');  sc.push('d');  Stack<char> sc1 = sc;  while(!si.isEmpty()) | {  cout << sd.top() << " " << si.top() << " " << sc1.top(); //display element at top    si.pop();  sd.pop();  sc1.pop();  }  } |

**Question # 2:**

Identify classes and attributes and Implement the following problem. Do assumptions where needed. **(20 points)**

There is a Dessert Shoppe which sells cookies by the dozen, ice cream, and sundaes (ice cream with a topping). Your code will be used for the checkout system.To do this, you will implement an inheritance hierarchy of classes derived from a **DessertItem** superclass.

**The DessertItem Class**

The **DessertItem** class is a *superclass* from which specific types of **DessertItems** can be derived. It contains only one data member, a name (char \*). It also defines a number of methods. All of the **DessertItem** class methods except the **getCost()** method are defined in a generic way.

The **getCost()** method in **DessertItem** class should only return 0 because the method of determining the costs varies based on the type of item.

Tax amounts should be rounded to the nearest Paisa. For example, calculating the tax on a food item with a cost of 199 paisa with a tax rate of 2.0% should be 4 paisa.

**The Derived Classes**

All of the classes which are derived from the **DessertItem** class must define a constructor.The **Cookie** class should be derived from the **DessertItem** class. A **Cookie** item has a *number* and a *price per* *dozen* which are used to determine its *cost*. For example, 4 cookies @ 399 paisa /dz. = 133 paisa. The costshould be rounded to the nearest paisa.

The **IceCream** class should be derived from the **DessertItem** class. An **IceCream** item simply has a *cost*.

The **Sundae** class should be derived from the **IceCream** class. The *cost* of a Sundae is the *cost of the IceCream* plus the *cost of the topping*.

**Make the two functions getCost() getTax() virtual in base class.**

Add attributes/functions where needed to implement this system.

In main user should be able to add items to a customer list. You need to keep track of list of items and your program should return total cost of a bill including tax.

|  |
| --- |
| int main()  {  int itemCount = 0;  DessertItem\* itemList[20];  //20 pointers of base class type, at most 20 objects can be added  char addMore;  char userChoice;  cout<<"Want to add more items\n"; cin>> addMore; //enter y for yes while(addMore ==’y’)  {  cout<<"Choose Item you want to add\n"; if(userChoice==1)  {  //take parameters from user needed for a Cookie  itemList[itemCount++]= new Cookie(...)  //making base class pointer //point cookie object  }  else if(userChoice==2)  {  //take parameters from user needed for an IceCream  itemList[itemCount++]= new IceCream(...)  //making base class pointer //point IceCream object  }  else if(userChoice==2)  {  //take parameters from user needed for a Sundae  itemList[itemCount++]= new Sundae(...)  //making base class pointer //point Sundae object  }  cout<<"Want to add more items\n"; cin>> addMore;  }  int TotalCost;  //Find totalCost (sum of costs of all items in list without tax) and print it.  int TotalTax;  //Find totalCost (sum of taxes of all items in list) and print it.  } |

**Question # 3: (15 points)**

A palindrome is a word, phrase, [number](https://en.wikipedia.org/wiki/Palindromic_number), or other sequence of [characters](https://en.wikipedia.org/wiki/Character_(symbol)) which reads the same backward as forward, such as 12321 or 52325.

Write a recursive function that return a Boolean value, receive an integer array, start and end index. It returns true, if array is palindrome otherwise false.