|  |  |
| --- | --- |
| Assignment 1 | |
| **Subject Name : Object Oriented Programming with JAVA**  **Subject Code : 2150704**  **Division : Computer Engineering**  **Coordinator : Hetal Gaudani**  **Submission Date : 20/7/15** | |
| **Sr. NO** | **Description** |
|  | Create a class student that has rollno, name, marks [5] as data member and void getinfo (), void putinfo () and float calCPI (). If student have marks less than 35 in any subjects then he is failed otherwise calculate CPI. And display student’s details. |
|  | Create a class Car that has Model no, Name, color & cost as Data member and create void getCarDetails () and void showCardDetails () as member function. Read details for 3 cars and display it. |
|  | Create a class BankAccount that has Depositor name , Acc\_no, Acc\_type, Balance as Data Members and void createAcc() . void Deposit(), void withdraw() and void BalanceInquiry as Member Function. When a new Account is created assign next serial no as account number. Account number starts from 1. |
|  | Create a class time that has hour, minute and second as data members. Create a parameterized constructor to initialize Time Objects. Create a member Function **Time Sum (Time, Time)** to sum two time objects. |
|  | Define a class Sales with the name and sales of salesman as data members. Calculate and print the Name, sales and commission, where commission id Rs .10 per 1000, if sales are at lease Rs. 25000 of more and Rs. 5 otherwise. Use appropriate member function. |
|  | Define a class with the Name, Basic salary and dearness allowance as data members.  Calculate and print the Name, Basic salary(yearly), dearness allowance and tax deduced at source(TDS) and net salary, where TDS is charged on gross salary which is basic salary + dearness allowance and TDS rate is as per following table.   |  |  | | --- | --- | | Gross Salary | TDS | | Rs. 100000 and below | NIL | | Above Rs. 100000 | 10% on excess over 100000 |   DA is 74% of Basic Salary for all  Use appropriate member function. |
|  | Define a class **Stack** with following members  **Fields:**  double array[];  int size  **Function:**  Stack(int size)  void push(double no)  boolean isEmpty( )  double pop( ) |
|  | Define a class **Array** with following member  **Field:**  int data[];  **Function:**  Array( ) //create array data of size 10  Array(int size) // create array of size **size**  Array(int data[]) // initialize array with parameter array  void Reverse \_an \_array () //reverse element of an array  int [Maximum \_of \_array](http://leepoint.net/notes-java/data/arrays/arrays-ex-max.html) () // find maximum element of array  [int Average\_of \_array](http://leepoint.net/notes-java/data/arrays/arrays-ex-mean.html)() //find average of element of array  void [Sorting](http://leepoint.net/notes-java/data/arrays/30arraysExamples.html) () //sort element of array  void display() //display element of array  int search(int no) //search element and return index else return -1  int size(); //return size of an array |
|  | Define a class **Matrix** with following  **Field:**  int row, column;  float mat[][]    **Function:**  Matrix(int a[][]);  Matrix();  void readMatrix() //read element of first array  float [][] transpose( ) //find transpose of first matrix  float [][] matrixMultiplication(Matrix second ) //multiply two matrices and return result  void displayMatrix(float [][]a) //display content of argument array  void displayMatrix() //display contne  void displayAnswerMatrix(float [][]a) // display content of answer array  int maximum\_of\_array() // return maximum element of first array  int average\_of\_array( ) // return average of first array    **create object of Matrix class in main and test all the functions in main** |
|  | Define class **Student** with following members  **Fields:**  String name  long enrollment\_no  float marks\_of\_cpu, marks\_of\_ds, marks\_of\_algo, marks\_of\_java  float percentage  **Functions:**  Student(String name, long enrollment\_no, float marks\_of\_cpu, float marks\_of\_ds, float marks\_of\_algo , float marks\_of\_java)  void read\_detail();  float calculate\_percentage( )  void display\_details( )  void display\_specific(String substring)  //above function display students details only if their name starts from substring or end with specify substring. For example display details of student whose surname is “Patel”.    Read the details of 5 students and display student details in ascending order of their percentage |
|  | Define a class **Cipher** with following data  **Field:**  String plainText;  int key  **Functions:**  Cipher(String plaintext,int key)  String Encryption( )  String Decryption( )  Read string and key from command prompt and replace every character of string with charcter which is key place down from current character.  Example  plainText = “GCET”  Key = 3  Encryption function written following String  “ JFHW”  Decryption function will convert encrypted string to original form  “GCET” |
|  | Define class **StringData** conatain following data  **Field:**  String data;  int len;  **Functions:**  StringData();  StringData(String a);  String reverse();  Boolean palindrome(String a);  int findSubstring(String subString) ; //Count the number of times one string is found in another.  int countWord();  int countCharacter();  int find(String substring) //find substring in main String and return index of first occurance of substring in main String  String replace(String find, String replace) //replace find with replace string |
|  | A bookshop maintains the inventory of books that are being sold at the shop. The list includes details such as author, title, price, publisher and stock position. When a customer wants a book, the sales person inputs the title or author and system search list and display whether it is available or not . If it is not, an appropriate message is displayed. Otherwise the message “required copies are available, the total cost of the requested copies is displayed; otherwise the message “required copies not in stock” is displayed.  The price of book should be updated as and when required.  The number of successful and unsuccessful transaction should be recorded for purpose of statistical analysis. |