**15-11-2022**

1. Take array value through keyboards as a integer and display sum of all number, sum even number, sum of odd numbers.
2. Take string array value through keyboards and display all names.
3. Take array value through keyboards as a integer and display those number in ascending order or descending order. (nested loop)
4. Create Employee class which contains three instance variable ie id,name,salary and two non static method calculateSalary() and dispalyEmployeeInfo() method.

In calcualteSalary method you need to declare three local variable as hra, da, and pf

And calculate the salary

Salary = salary+ 10% hra on salary + 5% da – 7% pf

In displayEmployeeInfo method display id,name,salary

In main class create two object and set the id,name,salary for one employee don’t call calculteSalary and second employee call both methods.

1. Create Employee class which contains three instance variable ie id,name,salary, write two constructor one is empty with default id,name,salary(123,unknown, 8000), parameterized constructor, and setValue Method, and two non static method calculateSalary() and dispalyEmployeeInfo() method.

In calcualteSalary method you need to declare three local variable as hra, da, and pf

And calculate the salary

Salary = salary+ 10% hra on salary + 5% da – 7% pf

In displayEmployeeInfo method display id,name,salary

In main class create 3 object

1st object empty constructor

2nd object parameterized constructor

3rd object set value through methods.

Then call display method for all objects.

**16-11-2022**

1. Take number of employee records using array as id,name,salary, designation. Then if desg is manager then give 5000, if designation is programmer then give 3000 else 1500 bonus

Then display all record ie id,name,salary,designation.

1. Make complete is and has relationship assignment.
2. Using method overloading do small example. Find the area of triangle, rectangle and circle and method name must be area and class may be Operation.
3. Please go through all example of method overriding, abstract and static example.

17-11-2022

1. create interface Bank which contains four abstract method ie

final int SIZE=10;

public String createAccount(int accno, String name, float amount);

public String withdrawAmount(int accno, float amount);

public String depositeAmount(int accno,float amount);

public String checkBalance(int accno);

createAccount, depositAmount, withdrawAmount, checkBalance

Create BankService class and that class implement Bank interface and it must be provide the body for all four methods.

This class contains three array variable of type accno, name and amount.

Static Count variable to keep the track how many account created.

public String createAccount(int accno, String name, float amount)

account number must be unique.

Min balance must be 500

public String withdrawAmount(int accno, float amount) {

if we give wrong account number then return account not exists

if correct we can display amount withdraw.

Maintain min 500.

}

public String depositAmount(int accno, float amount) {

if we give wrong account number then return account not exists

if correct we can display amount deposit.

We can’t deposit more than 50000

}

public String checkBalance(int accno) {

if we give wrong account number then return account not exists

return the balance.

}

Main class which contains main method

Do {

1: create accnount, 2:withdraw , 3: deposit 4: check balance

switch() {

1. pass the value for service class method

Break;

2

3

4

Default : wrong choice

}

Do you want to continue ?

}while();

18-11-2022

Inheritance, Interfaces and package.

|  |  |
| --- | --- |
| **Sr. No.** | **Assignment Question** |
| 1 | Australian Cricket is still the best team in the world and it has proved its supremacy in the last two World Cups held in the successive years 1999 and 2003.  The department handling the finance of the players is planning to develop software that would automatically calculate the income of the players based on their respective grade, the number of matches each player plays and also his performance in the tournament. Also, appropriate income tax should be applied on the income. To accomplish this, a team of experts have been chosen by the Australian Cricket Board to provide a solution for the same. Consider yourself to be a part of the team that implements the solution for designing the application.  Create an application using inheritance and interfaces to implement the Software. The application should consist of the following files.   1. Player.java 2. Tax.java bean package 3. PlayerIncome.java 4. GradeBonus.java 5. TournamentIncome.java service package 6. PlayerTest.java main package   Each file has a specific purpose and functionality. The descriptions of each file are as follows:  **Player.java**  The Player class is an abstract base class that contains an instance variable to store the name of the Player, and an abstract method.  The Booking class should contain an instance variable **name** to store the player’s name.  An abstract method named **displayDetails()** is created to display the name of the player.  **Tax.java**  An interface named **Tax** should be created that will act as an interface to calculate the tax of the players income.  Declare and initialize a constant variable **TAX\_PERCENT** to store the tax percentage. Also create an abstract method named **calculateTax()** to calculate the tax.  **PlayerIncome.java**  The PlayerIncome class should be a derived class of class Player and implement the interface Tax. This class should be used to store the Player details and display the same. |

|  |  |
| --- | --- |
|  | The PlayerIncome class should contain an instance variable **income** to store the player’s income.  Create a new instance of class PlayerIncome and store the salary in the instance variable and call the base constructor.  Now, create a method **calculateTax()** to calculate the tax on respective Player’s income.  Override the method **displayDetails()** to display the details about each player.  **GradeBonus.java**  Again an interface named **GradeBonus** is created to calculate the Bonus given to the player based on his grade.  Declare and initialize a constant variable **GRADE\_BONUS\_PERCENT** to store the grade bonus percentage. Also create an abstract method named **calculateGradeBonus()** to calculate the bonus to be given to a player on the basis of the grade he possess.  **TournamentIncome.java**  Again, create a derived class of the Player class and implement the GradeBonus interface. It should be used to store the details of income of each player for each tournament and display the same.  The TournamentIncome class should contain two instance variables **grade** and **rate** to store the player’s grade and the rate of each player per match he plays.  Create a new instance of class TournamentIncome and store the details in the three parameters **name**, **grade** and **rate**.  Now, create a method **calculateGradeBonus()** to calculate the bonus of each player based on his grade.  Create a method **displayDetails()** to display the details about each player. Here, calculate the income of the player using his income and the bonus amount he will be given as per his grade.  **PlayerTest.java**  Finally, create the main class PlayerTest that demonstrates the use of abstract classes and interfaces using the classes PlayerIncome and TournamentIncome. Create an object of PlayerIncome class and an object of TournamentIncome class and pass the argument. Now, display the details of  each employee of both the classes using the **displayDetails**() method. |