OODP TUTORIAL CLASS ASSIGNMENT

Keerthi suresh RA2411042010072

1. You’re developing a vehicle management system with a class Vehicle that has several data members (for example, an integer speed and a float fuelLevel) and a member function displayStatus(). Parts of the system need to dynamically modify and display these attributes using pointers to class members.

**#include <iostream>**

**using namespace std;**

**class Vehicle{**

**public:**

**int speed =10;**

**float fuellvl=20;**

**int vehicleNo=1423;**

**int displayStatus(){**

**cout<<"VEHICLE NO.: "<<vehicleNo<<"\n"<<"FUEL LEVEL: "<<fuellvl<<"\n"**

**<<"SPEED: "<<speed<<endl;**

**return 0;**

**}**

**};**

**int main(){**

**Vehicle obj;**

**int Vehicle::\*ptr1= &Vehicle::speed;**

**int (Vehicle::\*ptr2)() = &Vehicle::displayStatus;**

**obj.\*ptr1=60;**

**(obj.\*ptr2)();**

**return 0;**

**}**

Two common problems associated with pointers are undefined behaviour and garbage values. Undefined behavior occurs when two pointers are pointing to the same memory location and we free one of the pointers. Garbage values in the memory are those which cannot be accessed

2. Imagine you are designing a Student class. In this class, the student’s name should be directly accessible (public), but the id should be private and accessible only through a getter function.

**#include<iostream>**

**#include <string>**

**using namespace std;**

**class Student{**

**private:**

**int id = 2099813 ;**

**public:**

**int get\_id(){**

**return id;**

**}**

**};**

**int main(){**

**Student a;**

**string name;**

**cout<<"student name: "<<endl;**

**cin>>name;**

**cout<<"student id: "<<a.get\_id();**

**return 0;**

**}**

There might be some information that is known to some and not know to some, hence depending on the confidentiality ,a designer choose to allow direct access to some members while restricting others.

3. Design a C++ class named BankAccount with the following attributes:

accountNumber (integer)

balance (double)

accountHolderName (string)

Methods:

deposit(amount): Adds the specified amount to the balance.

withdraw(amount): Deducts the specified amount from the balance, if sufficient funds are available.

displayBalance(): Displays the current balance.

Write a program that:

Creates a BankAccount object.

Prompts the user to enter account details and initial balance.

Allows the user to perform deposit and withdrawal operations.

Displays the final balance

**#include <iostream>**

**#include <string>**

**using namespace std;**

**class BankAccount{**

**public:**

**int accountNumber;**

**double balance;**

**string accountHolderName;**

**void deposit(double amt){**

**if (amt>0){**

**balance+=amt;**

**cout<<"deposit of "<<amt<<"\n"<<"new balance: "<<balance<<endl;}**

**else{**

**cout<<"Invalid deposit amount"<<endl;**

**}**

**}**

**void withdraw(double amt){**

**if(amt>0 && amt <= balance){**

**balance-=amt;**

**cout<<"withdrawal of "<<amt<<"\n"<<"new balance: "<<balance<<endl;**

**}**

**else if(amt<=0){**

**cout<<"Invalid withdrawal amount"<<endl;**

**}**

**else{**

**cout<<"Insufficient funds"<<endl;**

**}**

**}**

**};**

**int main(){**

**BankAccount a;**

**cout<<"Welcome!"<<endl;**

**cout<<"enter account Holder Name: "<<endl;**

**cin>>a.accountHolderName;**

**cout<<"enter account Number: "<<endl;**

**cin>>a.accountNumber;**

**cout<<"enter balance: "<<endl;**

**cin>>a.balance;**

**int opts;**

**double amt;**

**cout<<"choose an option: \n"<<"1.depsoit\n"<<"2.withdraw\n"<<"3.exit"<<endl;**

**cout<<"enter your choice: "<<endl;**

**cin>>opts;**

**switch(opts){**

**case 1:**

**cout<<"enter deposit amount: "<<endl;**

**cin>>amt;**

**a.deposit(amt);**

**break;**

**case 2:**

**cout<<"enter withdrawal amount: "<<endl;**

**cin>>amt;**

**a.withdraw(amt);**

**break;**

**case 3:**

**cout<<"exiting.."<<endl;**

**break;**

**default:**

**cout<<"Invalid.please try again.";**

**}**

**cout<<"do visit again!";**

**return 0;**

**}**

4. Design a C++ class named Book with the following attributes:

title (string)

author (string)

isbn (string)

available (boolean)

Design a class named Library with the following attributes:

books (an array or vector of Book objects)

Methods:

addBook(book): Adds a new book to the library.

searchBookByTitle(title): Searches for a book based on its title.

borrowBook(isbn): Marks a book as unavailable if it's available.

returnBook(isbn): Marks a book as available.

**#include <iostream>**

**#include <string>**

**using namespace std;**

**class Book{**

**string title;**

**stringauthor;**

**string isbn;**

**bool available;**

**};**

**class Library{**

**};**