Experiment-8 Friend function and Static members

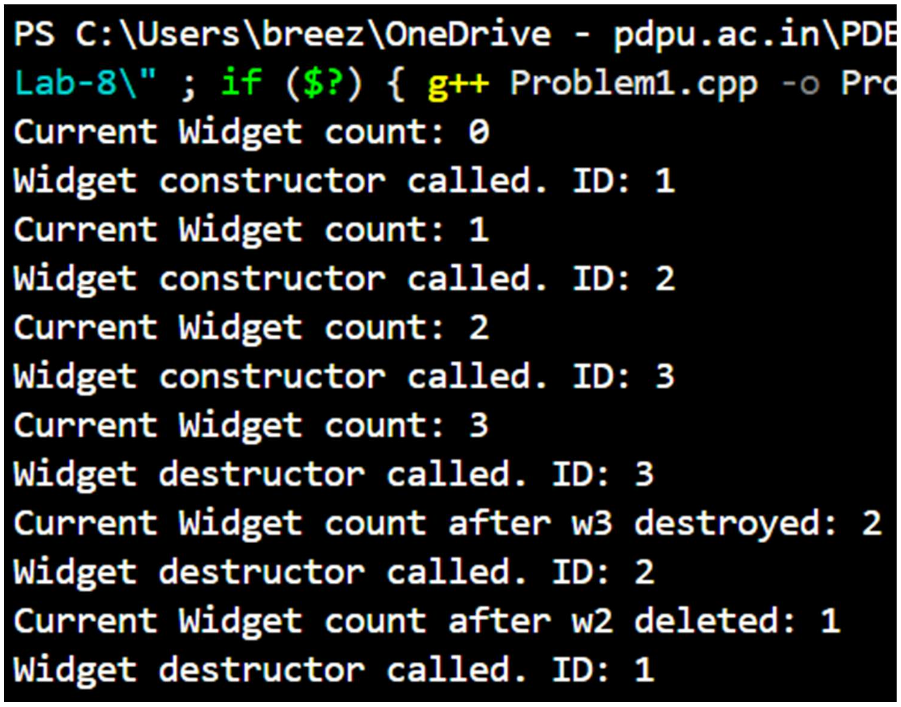
Problem 1: Use static variable to count number of objects.

Code:

```
#include <iostream>
using namespace std;
class Widget {
private:
   int id;
   static int objectCount;
   static int nextId;
public:
   Widget() {
      id = ++nextId;
      objectCount++;
      cout << "Widget constructor called. ID: " << id << endl;
   }
   ~Widget() {
      cout << "Widget destructor called. ID: " << id << endl;
      objectCount--;
   }
   static int getObjectCount() {
      return objectCount;
   }
};
int Widget::objectCount = 0;
int Widget::nextId = 0;
int main() {
   cout << "Current Widget count: " << Widget::getObjectCount() << endl;
   Widget w1;
```

```
    cout << "Current Widget count: " << Widget::getObjectCount() << endl;
    Widget* w2 = new Widget();
    cout << "Current Widget count: " << Widget::getObjectCount() << endl;
    {
        Widget w3;
        cout << "Current Widget count: " << Widget::getObjectCount() << endl;
    }
    cout << "Current Widget count after w3 destroyed: " << Widget::getObjectCount() <<
endl;
    delete w2;
    cout << "Current Widget count after w2 deleted: " << Widget::getObjectCount() << endl;
    return 0;
}
```

Output:



```
PS C:\Users\breez\OneDrive - pdpu.ac.in\PDE
Lab-8\" ; if ($?) { g++ Problem1.cpp -o Pro
Current Widget count: 0
Widget constructor called. ID: 1
Current Widget count: 1
Widget constructor called. ID: 2
Current Widget count: 2
Widget constructor called. ID: 3
Current Widget count: 3
Widget destructor called. ID: 3
Current Widget count after w3 destroyed: 2
Widget destructor called. ID: 2
Current Widget count after w2 deleted: 1
Widget destructor called. ID: 1
```

Problem 2: Write a C++ program to demonstrate the use of a friend function that operates on data from two different classes.

Code:

```
#include <iostream>
```

```
using namespace std;
class BankAccount;
class Wallet {
private:
   int cashAmount;
public:
   Wallet(int cash) : cashAmount(cash) {}
   void display() {
      cout << "My Wallet: Cash Amount = $" << cashAmount << endl;
   }
   friend int getTotalFunds(const Wallet& w, const BankAccount& ba);
};
class BankAccount {
private:
   int savings;
public:
   BankAccount(int s) : savings(s) {}
   void display() {
      cout << "My Bank Account: Savings = $" << savings << endl;
   }
   friend int getTotalFunds(const Wallet& w, const BankAccount& ba);
};
int getTotalFunds(const Wallet& w, const BankAccount& ba) {
   return w.cashAmount + ba.savings;
}
int main() {
   Wallet myWallet(500);
   BankAccount myAccount(1500);
   myWallet.display();
```

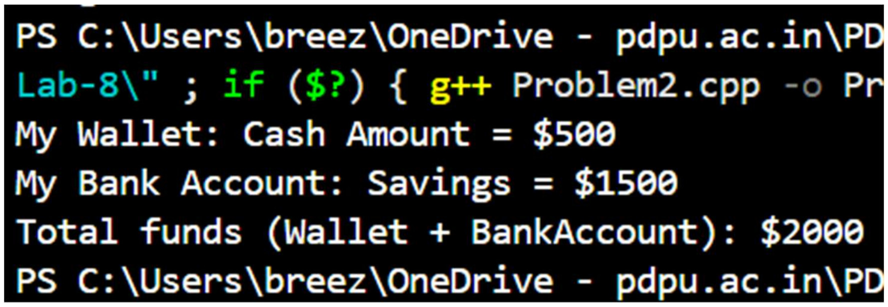
    myAccount.display();

    cout << "Total funds (Wallet + BankAccount): $" << getTotalFunds(myWallet, myAccount) << endl;

    return 0;

}

Output:



```
PS C:\Users\breez\OneDrive - pdpu.ac.in\PD
Lab-8\" ; if ($?) { g++ Problem2.cpp -o Pr
My Wallet: Cash Amount = $500
My Bank Account: Savings = $1500
Total funds (Wallet + BankAccount): $2000
PS C:\Users\breez\OneDrive - pdpu.ac.in\PD
```

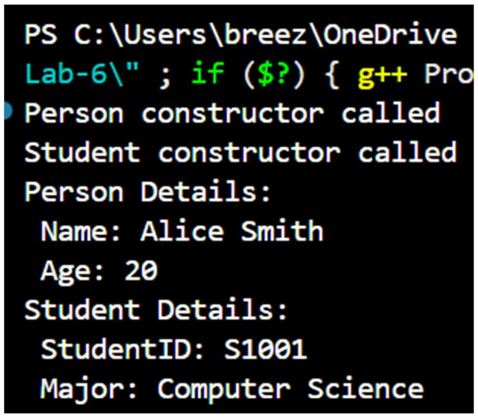
Experiment-6 Inheritance

Problem 1: Write a C++ program to demonstrate single inheritance using Person and Student classes.

Code:

```
#include<iostream>
using namespace std;

class Person
{
  private:
    string name;
    int age;
  public:
    Person(string n,int a)
    {
      name=n;
      age=a;
      cout<<"Person constructor called\n";
    }
    void displayPerson()
    {
      cout<<"Person Details:\n"<<" Name: "<<name<<"\n Age: "<<age<<"\n";
    }
};

class Student:public Person
{
  private:
    string StudentId;
    string major;
  public:
    Student(string name,int age,string Id,string maj):Person(name,age)
    {
      StudentId=Id;
      major=maj;
      cout<<"Student constructor called\n";
    }
    void displayStudent()
    {
      displayPerson();
      cout<<"Student Details:\n"<<" StudentID: "<<StudentId<<"\n Major: "<<major<<"\n";

    }
};

int main()
{
  Student student1("Alice Smith", 20, "S1001", "Computer Science");
  student1.displayStudent();
}
```

Output:



```
PS C:\Users\breez\OneDrive
Lab-6\" ; if ($?) { g++ Pro
Person constructor called
Student constructor called
Person Details:
 Name: Alice Smith
 Age: 20
Student Details:
 StudentID: S1001
 Major: Computer Science
```

Problem 2: Write a C++ program to illustrate the usage of this pointer and base class member access.

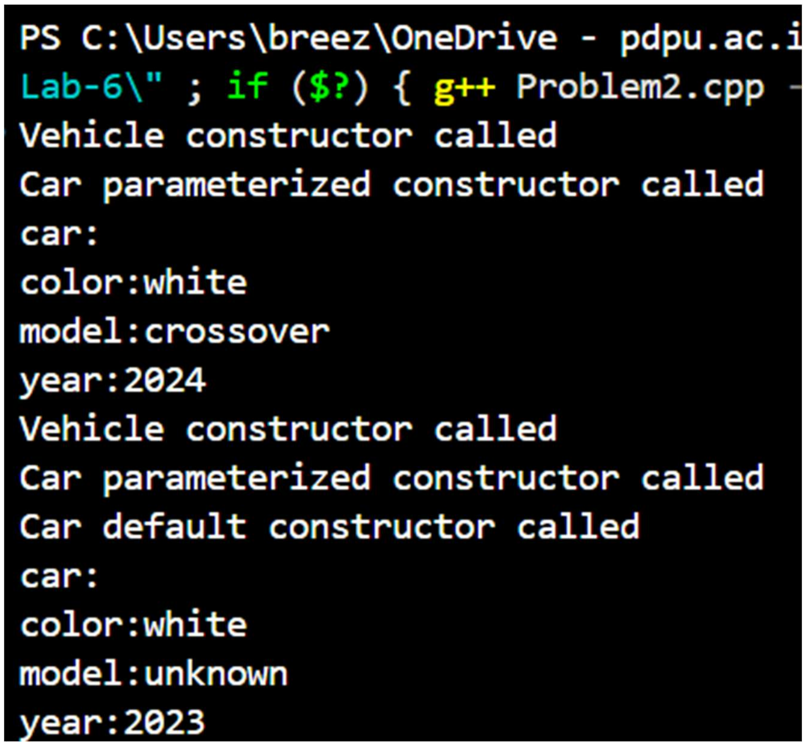
Code:

```
#include <iostream>
#include <string>
using namespace std;

class Vehicle
{
protected:
   string color;
public:
   Vehicle(string c): color(c)
   {
      cout << "Vehicle constructor called\n";
   }

   void displayColor()
   {
      cout << "color:" << color << "\n";
   }
};

class Car : public Vehicle
{
   string model;
   int year;
public:
   Car(string c, string m, int y): Vehicle(c)
```

```
    {
        this->model = m;
        this->year = y;
        cout << "Car parameterized constructor called\n";
    }

    Car(): Car("white", "unknown", 2023)
    {
        cout << "Car default constructor called\n";
    }

    void displayCarDetails()
    {
        cout << "car:\n";
        Vehicle::displayColor();
        cout << "model:" << this->model << "\n";
        cout << "year:" << this->year << "\n";
    }
};

int main()
{
    Car c1("white", "crossover", 2024);
    c1.displayCarDetails();

    Car c2;
    c2.displayCarDetails();

    return 0;
```

}

Output:



```
PS C:\Users\breez\OneDrive - pdpu.ac.i
Lab-6\" ; if ($?) { g++ Problem2.cpp -
Vehicle constructor called
Car parameterized constructor called
car:
color:white
model:crossover
year:2024
Vehicle constructor called
Car parameterized constructor called
Car default constructor called
car:
color:white
model:unknown
year:2023
```