4/8/2023

Object-Oriented Programming

Official Assignment-2

Abdullah Shafiq

22k-4489

Question-1

#include <iostream>

#include <string>

#include <cstdlib>

#include <stdlib.h>

#include <windows.h>

using namespace std;

class juice

{

int price;

string ingredient;

string taste;

public:

int getPrice()

{

return price;

}

void setPrice(int price)

{

this->price = price;

}

string getIngredient()

{

return ingredient;

}

void setIngredient(string ingredient)

{

this->ingredient = ingredient;

}

string getTaste()

{

return taste;

}

void setTaste(string taste)

{

this->taste = taste;

}

juice() {}

juice(int price, string ingredient, string taste)

{

setPrice(price);

setIngredient(ingredient);

setTaste(taste);

}

virtual void print()

{

cout << "=Price:" << getPrice() << endl;

cout << "=Ingredient:" << getIngredient() << endl;

cout << "=Taste:" << getTaste() << endl;

}

};

class fruit : public juice

{

protected:

string season;

public:

static int counter;

fruit() {}

fruit(int price, string ingredient, string taste) : juice(price, ingredient, taste)

{

}

};

int fruit::counter = 0;

class vegitables : public juice

{

string origin;

public:

static int counter;

vegitables() {}

vegitables(int price, string ingredient, string taste) : juice(price, ingredient, taste)

{

}

};

int vegitables::counter = 0;

class mixjuice : public juice

{

public:

static int counter;

mixjuice() {}

mixjuice(int price, string ingredient, string taste) : juice(price, ingredient, taste)

{

}

};

int mixjuice::counter = 0;

class citrus : public fruit

{

public:

citrus()

{

}

citrus(int price, string ingredient, string taste) : fruit(price, ingredient, taste)

{

}

void set\_citrus()

{

setPrice(99);

setIngredient("orange");

setTaste("sweet");

}

void print() override

{

cout << "\*Citrus Juice\*" << endl;

juice::print();

}

};

class tropical : public fruit

{

public:

tropical()

{

}

tropical(int price, string ingredient, string taste) : fruit(price, ingredient, taste)

{

}

void set\_tropical()

{

setPrice(99);

setIngredient("banana");

setTaste("sweet");

}

void print() override

{

cout << "\*Tropical Juice\*" << endl;

juice::print();

}

};

class berry : public fruit

{

public:

berry()

{

}

berry(int price, string ingredient, string taste) : fruit(price, ingredient, taste)

{

}

void set\_berry()

{

setPrice(99);

setIngredient("strawberry");

setTaste("sweet");

}

void print() override

{

cout << "\*Berry Juice\*" << endl;

juice::print();

}

};

class leefy : public vegitables

{

public:

leefy()

{

}

leefy(int price, string ingredient, string taste) : vegitables(price, ingredient, taste)

{

}

void set\_leefy()

{

setPrice(99);

setIngredient("spinach");

setTaste("sweet");

}

void print() override

{

cout << "\*Leefy Juice\*" << endl;

juice::print();

}

};

class root : public vegitables

{

public:

root()

{

}

root(int price, string ingredient, string taste) : vegitables(price, ingredient, taste)

{

}

void set\_root()

{

setPrice(99);

setIngredient("carrot");

setTaste("sweet");

}

void print() override

{

cout << "\*Root Juice\*" << endl;

juice::print();

}

};

class mixed : public vegitables

{

public:

mixed()

{

}

mixed(int price, string ingredient, string taste) : vegitables(price, ingredient, taste)

{

}

void set\_mixed()

{

setPrice(99);

setIngredient("beetroot");

setTaste("sweet");

}

void print() override

{

cout << "\*Mixed Juice\*" << endl;

juice::print();

}

};

class fruit\_veggy : public mixjuice

{

public:

fruit\_veggy()

{

}

fruit\_veggy(int price, string ingredient, string taste) : mixjuice(price, ingredient, taste)

{

}

void set\_fruit\_veggy()

{

setPrice(99);

setIngredient("apple");

setTaste("sweet");

}

void print() override

{

cout << "\*Fruit-Veggy Juice\*" << endl;

juice::print();

}

};

class smoothey : public mixjuice

{

public:

smoothey()

{

}

smoothey(int price, string ingredient, string taste) : mixjuice(price, ingredient, taste)

{

}

void set\_smoothey()

{

setPrice(99);

setIngredient("mango");

setTaste("sweet");

}

void print() override

{

cout << "\*Smoothey Juice\*" << endl;

juice::print();

}

};

class sales : public juice

{

int saleID;

string customerName;

int totalBill;

int numberOfFruitJuice;

int numberOfVegetableJuice;

int numberOfMixedJuice;

public:

int bill = 0;

sales() {}

sales(int saleID, string customerName, int totalBill) : juice()

{

setSaleID(saleID);

setCustomerName(customerName);

setTotalBill(totalBill);

}

void setSaleID(int saleID)

{

this->saleID = saleID;

}

void setCustomerName(string customerName)

{

this->customerName = customerName;

}

void setTotalBill(int totalBill)

{

this->totalBill = totalBill;

}

int getSaleID()

{

return saleID;

}

string getCustomerName()

{

return customerName;

}

int getTotalBill()

{

return totalBill;

}

int getNumberOfFruitJuice()

{

return fruit::counter;

}

int getNumberOfVegetableJuice()

{

return vegitables::counter;

}

int getNumberOfMixedJuice()

{

return mixjuice::counter;

}

int getTbill()

{

bill = (fruit::counter \* 99) + (vegitables::counter \* 99) + (mixjuice::counter \* 99);

return bill;

}

void print()

{

cout<<"\n### 22k-4489 ###\n### Abdullah Shafiq ###"<<endl;

cout << "Sale ID:" << getSaleID() << endl;

cout << "Customer Name:" << getCustomerName() << endl;

cout << "Total Bill:" << getTbill() << endl;

cout << "The Total Number Fruit Juices: " << getNumberOfFruitJuice() << endl;

cout << "The Total Number Vegetable Juices: " << getNumberOfVegetableJuice() << endl;

cout << "The Total Number Mixed Juices: " << getNumberOfMixedJuice() << endl;

}

friend sales operator+(const sales &sal, const sales &sal1);

};

sales operator+( sales &s1,sales &s2)

{

if (s1.getCustomerName() != s2.getCustomerName())

{

throw invalid\_argument("Customer names do not match");

}

sales result;

int bill;

result.setCustomerName(s1.getCustomerName());

result.setSaleID(rand());

result.setTotalBill(s1.getTbill() + s2.getTbill());

return result;

}

int main()

{

citrus c;

tropical t;

berry b;

leefy l;

root r;

mixed m;

fruit\_veggy fv;

smoothey s;

sales sal, sal1;

string customer\_name;

string newname;

int option;

cout<<"\n### 22k-4489 ###\n### Abdullah Shafiq ###"<<endl;

here:

cout << "Welcome to our juice shop! What is your name?\n";

cin >> customer\_name;

if (sal.getCustomerName() == customer\_name)

{

cout << "You are already in our system!\n";

sal1.setCustomerName(customer\_name);

}

else if (sal.getCustomerName() != customer\_name)

{

cout << "Welcome to our juice shop! Can you please enter your name again?\n";

cin >> newname;

sal1.setCustomerName(newname);

fruit::counter = 0;

vegitables::counter = 0;

mixjuice::counter = 0;

sal.bill = 0;

}

cout << "Hello, " << customer\_name << "<3\n";

cout << "Here is our menu:\n";

cout << "1-Fruit Juices" << endl;

cout << "2-Vegetable Juices" << endl;

cout << "3-Mixed Juices" << endl;

cout << "4-Checkout" << endl;

cout << "5-Exit from menu" << endl;

cout << "6-Exit from main program\n\n";

cout << "Enter your option: ";

cin >> option;

while (option)

{

system("cls");

cout<<"\n### 22k-4489 ###\n### Abdullah Shafiq ###"<<endl;

switch (option)

{

case 1:

fruit::counter++;

cout << "1-Citrus Juice" << endl;

cout << "2-Tropical Juice" << endl;

cout << "3-Berry Juice" << endl;

cout << "Enter your option: ";

cin >> option;

switch (option)

{

case 1:

sal.setTotalBill(99);

c.set\_citrus();

c.print();

break;

case 2:

sal.setTotalBill(99);

t.set\_tropical();

t.print();

break;

case 3:

sal.setTotalBill(99);

b.set\_berry();

b.print();

break;

default:

cout << "Invalid Option" << endl;

break;

}

break;

case 2:

vegitables::counter++;

cout << "1-Leefy Juice" << endl;

cout << "2-Root Juice" << endl;

cout << "3-Mixed Juice" << endl;

cout << "Enter your option: ";

cin >> option;

switch (option)

{

case 1:

sal.setTotalBill(99);

l.set\_leefy();

l.print();

break;

case 2:

sal.setTotalBill(99);

r.set\_root();

r.print();

break;

case 3:

sal.setTotalBill(99);

m.set\_mixed();

m.print();

break;

default:

cout << "Invalid Option" << endl;

break;

}

break;

case 3:

mixjuice::counter++;

cout << "1-Fruit-Veggy Juice" << endl;

cout << "2-Smoothey Juice" << endl;

cout << "Enter your option: ";

cin >> option;

switch (option)

{

case 1:

sal.setTotalBill(99);

fv.set\_fruit\_veggy();

fv.print();

break;

case 2:

sal.setTotalBill(99);

s.set\_smoothey();

s.print();

break;

default:

cout << "Invalid Option" << endl;

break;

}

break;

case 4:

sal.setSaleID(rand());

sal.setCustomerName(customer\_name);

sal.print();

break;

case 5:

goto here;

break;

case 6:

exit(0);

break;

default:

cout << "Invalid Option" << endl;

break;

}

cout << "\nMain Course:" << endl;

cout << "(1-Fruit Juices)" << endl;

cout << "(2-Vegetable Juices)" << endl;

cout << "(3-Mixed Juices)" << endl;

cout << "4-Checkout" << endl;

cout << "5-Exit" << endl;

cout << "6-Exit from main program" << endl;

cout << "0-Operator overloaded bill\n\n";

cout << "Enter your option: ";

cin >> option;

}

sales sal2 = sal + sal1;

sal2.print();

return 0;

}

Output:

### 22k-4489 ###

### Abdullah Shafiq ###

Welcome to our juice shop! What is your name?

Abdullah

Welcome to our juice shop! Can you please enter your name again?

Abdullah

Hello, Abdullah<3

Here is our menu:

1-Fruit Juices

2-Vegetable Juices

3-Mixed Juices

4-Checkout

5-Exit from menu

6-Exit from main program

Enter your option:1 **// entered option 1 for Fruit Juice**

1-Citrus Juice

2-Tropical Juice

3-Berry Juice

Enter your option: 1

\*Citrus Juice\*

=Price:99

=Ingredient:orange

=Taste:sweet

Main Course:

(1-Fruit Juices)

(2-Vegetable Juices)

(3-Mixed Juices)

4-Checkout

5-Exit

6-Exit from main program

0-Operator overloaded bill

Enter your option:4 **// Entered Option 4 For Checkout**

### 22k-4489 ###

### Abdullah Shafiq ###

Sale ID:41

Customer Name:Abdullah

Total Bill:99

The Total Number Fruit Juices: 1

The Total Number Vegetable Juices: 0

The Total Number Mixed Juices: 0

Main Course:

(1-Fruit Juices)

(2-Vegetable Juices)

(3-Mixed Juices)

4-Checkout

5-Exit

6-Exit from main program

0-Operator overloaded bill

Enter your option:5 **// Entered Option 5 for exit just from menu**

Welcome to our juice shop! What is your name?

Abdullah **// Entered Same name again for matching**

You are already in our system!

Hello, Abdullah<3

Here is our menu:

1-Fruit Juices

2-Vegetable Juices

3-Mixed Juices

4-Checkout

5-Exit from menu

6-Exit from main program

Enter your option:2 **// Entered Option 2 for vegitable juice**

1-Leefy Juice

2-Root Juice

3-Mixed Juice

Enter your option: 1

\*Leefy Juice\*

=Price:99

=Ingredient:spinach

=Taste:sweet

Main Course:

(1-Fruit Juices)

(2-Vegetable Juices)

(3-Mixed Juices)

4-Checkout

5-Exit

6-Exit from main program

0-Operator overloaded bill

Enter your option:4 **// Entered Option 4 fot again checkout**

Sale ID:18467

Customer Name:Abdullah

Total Bill:198

The Total Number Fruit Juices: 1

The Total Number Vegetable Juices: 1

The Total Number Mixed Juices: 0

Main Course:

(1-Fruit Juices)

(2-Vegetable Juices)

(3-Mixed Juices)

4-Checkout

5-Exit

6-Exit from main program

0-Operator overloaded bill

Enter your option:0 **// Entered Option 0 for operator overloading result**

Sale ID:6334

Customer Name:Abdullah

Total Bill:198

The Total Number Fruit Juices: 1

The Total Number Vegetable Juices: 1

The Total Number Mixed Juices: 0

PS C:\Users\surfac\OneDrive\Desktop\C++\OOP Assignment 02>

Question-2

#include <iostream>

#include <cstring>

#include <iomanip>

using namespace std;

class course

{

string c\_name;

string c\_code;

string c\_grade;

int c\_hours;

public:

string getC\_name()

{

return c\_name;

}

void setC\_name(string c\_name)

{

this->c\_name = c\_name;

}

string getC\_code()

{

return c\_code;

}

void setC\_code(string c\_code)

{

this->c\_code = c\_code;

}

int getC\_hours()

{

return c\_hours;

}

void setC\_hours(int c\_hours)

{

this->c\_hours = c\_hours;

}

string getC\_grade()

{

return c\_grade;

}

void setC\_grade(string c\_grade)

{

this->c\_grade = c\_grade;

}

course()

{

}

course(string c\_name, string c\_code, int c\_hours, string c\_grade)

{

setC\_name(c\_name);

setC\_code(c\_code);

setC\_hours(c\_hours);

setC\_grade(c\_grade);

}

};

class student : public course

{

string s\_id;

string s\_name;

double s\_fees = 0;

int c\_enrolled;

bool feespaid;

int t\_hours = 0;

double t\_grade = 0;

string grade;

int hour;

public:

string getS\_id()

{

return s\_id;

}

void setS\_id(string s\_id)

{

this->s\_id = s\_id;

}

string getS\_name()

{

return s\_name;

}

void setS\_name(string s\_name)

{

this->s\_name = s\_name;

}

int getC\_enrolled()

{

return c\_enrolled;

}

void setC\_enrolled(int c\_enrolled)

{

this->c\_enrolled = c\_enrolled;

}

void set\_fees\_paid(bool feespaid)

{

this->feespaid = feespaid;

}

bool get\_fees\_paid()

{

return feespaid;

}

student()

{

}

student(string s\_id, string s\_name, int c\_enrolled)

{

setS\_id(s\_id);

setS\_name(s\_name);

setC\_enrolled(c\_enrolled);

}

void cal\_t\_grade(string grade, int hour)

{

this->grade = grade;

if (hour == 3)

{

if (grade == "A")

{

t\_grade = t\_grade + (4.0 \* 3);

}

else if (grade == "A-")

{

t\_grade = t\_grade + (3.67 \* 3);

}

else if (grade == "B+")

{

t\_grade = t\_grade + (3.33 \* 3);

}

else if (grade == "B")

{

t\_grade = t\_grade + (3.0 \* 3);

}

else if (grade == "B-")

{

t\_grade = t\_grade + (2.67 \* 3);

}

else if (grade == "C")

{

t\_grade = t\_grade + (2.0 \* 3);

}

else if (grade == "D")

{

t\_grade = t\_grade + (1.0 \* 3);

}

else if (grade == "F")

{

t\_grade = t\_grade + 0.0;

}

}

else if (hour == 2)

{

if (grade == "A")

{

t\_grade = t\_grade + (4.0 \* 2);

}

else if (grade == "A-")

{

t\_grade = t\_grade + (3.67 \* 2);

}

else if (grade == "B+")

{

t\_grade = t\_grade + (3.33 \* 2);

}

else if (grade == "B")

{

t\_grade = t\_grade + (3.0 \* 2);

}

else if (grade == "B-")

{

t\_grade = t\_grade + (2.67 \* 2);

}

else if (grade == "C")

{

t\_grade = t\_grade + (2.0 \* 2);

}

else if (grade == "D")

{

t\_grade = t\_grade + (1.0 \* 2);

}

else if (grade == "F")

{

t\_grade = t\_grade + 0.0;

}

}

}

double getT\_grade()

{

return t\_grade;

}

void cal\_crh(int hr)

{

t\_hours = t\_hours + hr;

}

int getT\_hours()

{

return t\_hours;

}

void cal\_GPA()

{

float gpa = 0;

gpa = getT\_grade() / getT\_hours();

cout << "GPA: " << gpa << endl;

}

void cal\_PER()

{

float per;

per = ((getT\_grade() / getT\_hours()) \* 100) / 4.0;

cout << "Percentage: " << per << "%" << endl;

}

void cal\_fees(int id)

{

id = (id / 2) \* 1000;

if (get\_fees\_paid() == true)

{

s\_fees = getT\_hours() \* id;

}

else if (get\_fees\_paid() == false)

{

s\_fees = getT\_hours() \* id;

s\_fees = s\_fees + (s\_fees \* 0.05);

}

}

double getS\_fees()

{

return s\_fees;

}

};

int main()

{

string id;

string s\_name;

int c\_enrolled;

string c\_name;

string c\_code;

string c\_grade;

int c\_hours;

bool status;

int first\_2\_digits;

cout<<"\n### 22k-4489 ###\n### Abdullah Shafiq ###"<<endl;

cout << "Enter Student ID: ";

cin >> id;

cout << "Enter Student Name: ";

cin >> s\_name;

cout << "Enter Number of Courses Enrolled: ";

cin >> c\_enrolled;

student s(id, s\_name, c\_enrolled);

first\_2\_digits = stoi(id.substr(0, 2));

course c[c\_enrolled];

for (int i = 0; i < c\_enrolled; i++)

{

cout << "Enter Course Name: ";

cin >> c\_name;

cout << "Enter Course Code: ";

cin >> c\_code;

cout << "Enter Course Hours: ";

cin >> c\_hours;

cout << "Enter Course Grade: ";

cin >> c\_grade;

c[i].setC\_name(c\_name);

c[i].setC\_code(c\_code);

c[i].setC\_hours(c\_hours);

c[i].setC\_grade(c\_grade);

s.cal\_t\_grade(c\_grade, c\_hours);

s.cal\_crh(c\_hours);

}

cout << "Enter Fees Status (1/0): ";

cin >> status;

cout<<"\n### 22k-4489 ###\n### Abdullah Shafiq ###"<<endl;

cout << "Student ID: " << s.getS\_id() << endl;

cout << "Student Name: " << s.getS\_name() << endl;

cout << "Number of Courses Enrolled: " << s.getC\_enrolled() << endl;

if (status == true)

{

s.set\_fees\_paid(true);

s.cal\_fees(first\_2\_digits);

cout << "Course Name" << setw(15) << "Course Code" << setw(15) << "Credit Hours" << setw(15) << "Course Grade" << endl;

for (int k = 0; k < c\_enrolled; k++)

{

cout << c[k].getC\_name() << setw(18) << c[k].getC\_code() << setw(15) << c[k].getC\_hours() << setw(15) << c[k].getC\_grade() << endl;

}

// cout << "Grade: " << s.getT\_grade();

cout << "\nTotal Credit Hours :" << s.getT\_hours() << endl;

cout << "Fees \*PAID\* :" << s.getS\_fees() << endl;

s.cal\_GPA();

s.cal\_PER();

}

else if (status == false)

{

s.set\_fees\_paid(false);

s.cal\_fees(first\_2\_digits);

cout << "Course Name" << setw(15) << "Course Code" << setw(15) << "Credit Hours" << setw(15) << "Course Grade" << endl;

for (int k = 0; k < c\_enrolled; k++)

{

cout << c[k].getC\_name() << setw(18) << c[k].getC\_code() << setw(15) << c[k].getC\_hours() << setw(15) << "LOCKED" << endl;

}

cout << "\nTotal Credit Hours :" << s.getT\_hours() << endl;

cout << "Fees \*NOT PAID\*, Amount DUE :" << s.getS\_fees();

}

return 0;

}

Output:

When fees paid:

### 22k-4489 ###

### Abdullah Shafiq ###

Enter Student ID: 22k-4489

Enter Student Name: Abdullah

Enter Number of Courses Enrolled: 5

Enter Course Name: OOP

Enter Course Code: OOP1005

Enter Course Hours: 3

Enter Course Grade: A

Enter Course Name: DLD

Enter Course Code: DLD1005

Enter Course Hours: 3

Enter Course Grade: B+

Enter Course Name: DE

Enter Course Code: DE1005

Enter Course Hours: 3

Enter Course Grade: A

Enter Course Name: CPS

Enter Course Code: CPS1003

Enter Course Hours: 2

Enter Course Grade: B+

Enter Course Name: PST

Enter Course Code: PST1008

Enter Course Hours: 3

Enter Course Grade: A

Enter Fees Status (1/0): 1

### 22k-4489 ###

### Abdullah Shafiq ###

Student ID: 22k-4489

Student Name: Abdullah

Number of Courses Enrolled: 5

Course Name Course Code Credit Hours Course Grade

OOP OOP1005 3 A

DLD DLD1005 3 B+

DE DE1005 3 A

CPS CPS1003 2 B+

PST PST1008 3 A

Total Credit Hours :14

Fees \*PAID\* :154000

GPA: 3.76071

Percentage: 94.0179%

When fees not paid:

### 22k-4489 ###

### Abdullah Shafiq ###

Student ID: 22k-4489

Student Name: Abdullah

Number of Courses Enrolled: 5

Course Name Course Code Credit Hours Course Grade

DLD DLD1005 3 LOCKED

DES DES1006 3 LOCKED

OOP OOP1007 3 LOCKED

CPS SPS1004 2 LOCKED

PST PST1008 3 LOCKED

Total Credit Hours :14

Fees \*NOT PAID\*, Amount DUE :161700

PS C:\Users\surfac\OneDrive\Desktop\C++\OOP Assignment 02>

Question-3

#include <iostream>

#include <string>

using namespace std;

class Product

{

private:

string name;

float price;

public:

Product() {}

Product(string name, float price)

{

setName(name);

setPrice(price);

}

void setName(string name)

{

this->name = name;

}

void setPrice(int price)

{

this->price = price;

}

string getName()

{

return name;

}

float getPrice()

{

return price;

}

virtual float get\_discounted\_Price()

{

return price;

}

virtual void print\_details()

{

cout << "Name: " << getName() << endl;

cout << "Price: " << getPrice() << endl;

}

friend Product operator&( Product& p6, Product& p7);

};

Product operator&( Product& p6, Product& p7)

{

float avg;

avg = (p6.getPrice() + p7.getPrice()) / 2;

string name;

name = p6.getName() + " & " + p7.getName();

return Product(name, avg);

}

class Book : public Product

{

string author;

public:

Book() {}

Book(string name, float price, string author) : Product(name, price)

{

setAuthor(author);

}

void setAuthor(string author)

{

this->author = author;

}

string getAuthor()

{

return author;

}

void print\_details() override

{

cout << "Name: " << getName() << endl;

cout << "Price: " << getPrice() << endl;

cout << "Author: " << getAuthor() << endl;

}

float get\_discounted\_Price() override

{

return getPrice() - (getPrice() \* 0.05);

}

};

class Electronic : public Product

{

string brand;

public:

Electronic() {}

Electronic(string name, float price, string brand) : Product(name, price)

{

setBrand(brand);

}

void setBrand(string brand)

{

this->brand = brand;

}

string getBrand()

{

return brand;

}

void print\_details() override

{

cout << "Name: " << getName() << endl;

cout << "Price: " << getPrice() << endl;

cout << "Brand: " << getBrand() << endl;

}

float get\_discounted\_Price() override

{

return getPrice() - (getPrice() \* 0.15);

}

};

class Clothing : public Product

{

string size;

public:

Clothing() {}

Clothing(string name, float price, string size) : Product(name, price)

{

setSize(size);

}

void setSize(string size)

{

this->size = size;

}

string getSize()

{

return size;

}

void print\_details() override

{

cout << "Name: " << getName() << endl;

cout << "Price: " << getPrice() << endl;

cout << "Size: " << getSize() << endl;

}

float get\_discounted\_Price() override

{

return getPrice() - (getPrice() \* 0.1);

}

};

class Customer

{

string name;

float balance;

public:

Customer() {}

Customer(string name, float balance)

{

setName(name);

setBalance(balance);

}

void setName(string name)

{

this->name = name;

}

void setBalance(float balance)

{

this->balance = balance;

}

string getName()

{

return name;

}

float getBalance()

{

return balance;

}

void print\_details()

{

cout << "Name: " << getName() << endl;

cout << "Balance: " << getBalance() << endl;

}

virtual void buy(Product \*p)

{

if (getBalance() >= p->getPrice())

{

setBalance(getBalance() - p->getPrice());

cout << "You have bought " << p->getName() << endl;

cout << "Your balance is " << getBalance() << endl;

}

else

{

cout << "You don't have enough balance to buy " << p->getName() << endl;

}

}

};

class VIP\_customers : public Customer

{

public:

VIP\_customers() {}

VIP\_customers(string name, float balance) : Customer(name, balance) {}

void buy(Product \*p) override

{

if (getBalance() >= p->get\_discounted\_Price())

{

setBalance(getBalance() - p->get\_discounted\_Price());

cout << "You have bought " << p->getName() << endl;

cout << "Your balance is " << getBalance() << endl;

}

else

{

cout << "You don't have enough balance to buy " << p->getName() << endl;

}

}

};

int main()

{

cout<<"\n### 22k-4489 ###\n### Abdullah Shafiq ###"<<endl;

Product \*p1 = new Product("Product 1", 100);

Product \*p2 = new Book("Book 1", 200, "Author 1");

Product \*p3 = new Electronic("Electronic 1", 300, "Brand 1");

Product \*p4 = new Clothing("Clothing 1", 400, "Size 1");

Customer \*c1 = new Customer("Customer 1", 1000);

Customer \*c2 = new VIP\_customers("VIP Customer 1", 1000);

cout<<"\*\*\*Regular Customer Buying Item\*\*\*"<<endl;

c1->buy(p1);

c1->buy(p2);

c1->buy(p3);

c1->buy(p4);

cout<<"\n\*\*\*VIP Customer Buying Item\*\*\*"<<endl;

c2->buy(p1);

c2->buy(p2);

c2->buy(p3);

c2->buy(p4);

cout<<"\n### 22k-4489 ###\n### Abdullah Shafiq ###"<<endl;

// creating 2 products with the prices of 2 and 10.

cout<<"\nOperator & called:"<<endl;

Product p6("Book",10);

Product p7("Pen",2);

Product p8 = p6 & p7;

p8.print\_details();

return 0;

}

Output:

### 22k-4489 ###

### Abdullah Shafiq ###

\*\*\*Regular Customer Buying Item\*\*\*

You have bought Product 1

Your balance is 900

You have bought Book 1

Your balance is 700

You have bought Electronic 1

Your balance is 400

You have bought Clothing 1

Your balance is 0

\*\*\*VIP Customer Buying Item\*\*\*

You have bought Product 1

Your balance is 900

You have bought Book 1

Your balance is 710

You have bought Electronic 1

Your balance is 455

You have bought Clothing 1

Your balance is 95

### 22k-4489 ###

### Abdullah Shafiq ###

Operator & called:

Name: Book & Pen

Price: 6

PS C:\Users\surfac\OneDrive\Desktop\C++\OOP Assignment 02>

Question-4

#include <iostream>

#include <string>

using namespace std;

class gameObject

{

private:

string name;

int x;

int y;

public:

gameObject(){}

gameObject(string name, int x, int y)

{

setName(name);

setX(x);

setY(y);

}

void setX(int x)

{

this->x = x;

}

void setY(int y)

{

this->y = y;

}

int getX()

{

return x;

}

int getY()

{

return y;

}

void setName(string name)

{

this->name = name;

}

string getName()

{

return name;

}

virtual void draw()

{

cout << "Drawing " << getName() << " at " << getX() << ", " << getY() << endl;

}

};

class Player:public gameObject

{

int health;

public:

Player(){}

Player(string name, int x, int y, int health):gameObject(name, x, y)

{

setHealth(health);

}

void setHealth(int health)

{

this->health = health;

}

int getHealth()

{

return health;

}

void draw() override

{

cout << "Drawing " << getName() << " at " << getX() << ", " << getY() << endl;

cout<< "Health: "<<getHealth()<<endl;

}

friend bool operator==(Player& p1, Player& p2);

};

class Enemy:public gameObject

{

int demage;

public:

Enemy(){}

Enemy(string name, int x, int y, int demage):gameObject(name, x, y)

{

setDemage(demage);

}

void setDemage(int demage)

{

this->demage = demage;

}

int getDemage()

{

return demage;

}

void draw() override

{

gameObject::draw();

cout<< "Demage: "<<getDemage()<<endl;

}

};

class Powerup:public gameObject

{

int effect;

public:

Powerup(){}

Powerup(string name, int x, int y, int effect):gameObject(name, x, y)

{

setEffect(effect);

}

void setEffect(int effect)

{

this->effect = effect;

}

int getEffect()

{

return effect;

}

void draw() override

{

gameObject::draw();

cout<< "Effect: "<<getEffect()<<endl;

}

};

class Game

{

gameObject \*arr[3];

public:

Game(){}

Game(gameObject \*arr[3])

{

for(int i = 0; i < 3; i++)

{

this->arr[i] = arr[i];

}

}

void drawAll()

{

for(int i = 0; i < 3; i++)

{

arr[i]->draw();

}

}

};

bool operator==(Player& p1, Player& p2)

{

if(p1.getHealth() == p2.getHealth())

{

return true;

}

else

{

return false;

}

}

int main()

{

cout<<"\n### 22k-4489 ###\n### Abdullah Shafiq ###"<<endl;

Player p1("Player 1", 10, 20, 100);

Enemy e1("Enemy", 30, 40, 50);

Powerup pu1("Powerup", 50, 60, 70);

gameObject \*arr[3] = {&p1, &e1, &pu1};

Game g1(arr);

g1.drawAll();

// Creating player 2 form comapring health.

cout<<"\nComparing Health:"<<endl;

Player p2("Player 2", 10, 20, 200);

cout<<endl;

p1.draw();

p2.draw();

if(p1 == p2)

{

cout << "Both player have same health" << endl;

}

else

{

cout << "Both player have different health" << endl;

}

Player p3("Player 3", 10, 20, 100);

cout<<endl;

p1.draw();

p3.draw();

if(p1 == p3)

{

cout << "Both player have same health" << endl;

}

else

{

cout << "Both player have different health" << endl;

}

return 0;

}

Output:

### 22k-4489 ###

### Abdullah Shafiq ###

Drawing Player 1 at 10, 20

Health: 100

Drawing Enemy at 30, 40

Demage: 50

Drawing Powerup at 50, 60

Effect: 70

Comparing Health:

Drawing Player 1 at 10, 20

Health: 100

Drawing Player 2 at 10, 20

Health: 200

Both players have different health

Drawing Player 1 at 10, 20

Health: 100

Drawing Player 3 at 10, 20

Health: 100

Both players have same health

PS C:\Users\surfac\OneDrive\Desktop\C++\OOP Assignment 02>