

**Софийски Университет “Климент Охридски”**  
**Факултет по Математика и Информатика**

**Контролно No. 1b**

**Курс:** Обектно Ориентирано Програмиране с C#.NET

**Преподавател:** д-р. Е. Кръстев

**Студент :**

**Дата:** януари 9, 2021

**Време за работа:** 120 min

**Инструкции:** Изпълнете следното задание за обектно ориентирано програмиране и предайте пълния набор от файлове необходими за решаване на програмата на флопи диск. Пълен набор от точки се присъжда за пълно решение на съответната подзадача.

**Оценки:**

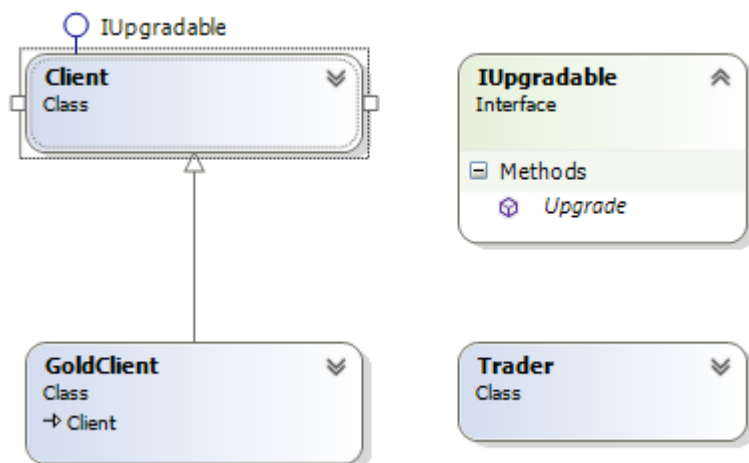
2	от 0 до 54 точки
3	от 55 до 64 точки
4	от 65 до 74 точки
5	от 75 до 84 точки
6	от 85 до 100 точки

**Задача 1 ( 100 точки)**

Приложете следните принципи на Обектно ориентираното програмиране:

- *hiding of information*
- *software reuse*
- *inheritance*
- *polymorphism*

и напишете **библиотека от класове, а после и конзолно приложение** на **C#.NET**, използващо тази библиотека за автоматизирано прилагане на **бизнес правила** при задаване на ранг на клиенти. При това се приема, че **търговецът (Trader)** има два ранга за своите клиенти-обикновени (Client) и „златни“ (GoldClient) клиенти в зависимост от средната стойност на направените покупки според дадената UML диаграма по-долу.



Всеки клиент (Client) е IUpgradable т.е. може да се преминава в по- висок ранг, при който клиентът може да ползва кредит. Допълнително, търговецът (Trader) има списък от клиенти, чиито ранг се определя по правила, които се задават в отделно приложение и така могат да се променят при нужда независимо от търговеца.

При реализацията на приложението извършете следните действия (*Използвайте отделни файлове за всеки class или interface. Използвайте означенията, въведени по-долу*)

А. . Създайте нов проект от тип клас библиотека и напишете в него следното (70 точки):

1. Напишете interface `IUpgradable`, който да има:

- void метод `Upgrade`, който има единствен аргумент от стандартен тип делегат `Action<IUpgradable>`.

Точки: 4

2. Напишете class `Client`, който има

- списък (тип `List`) `purchases` от стойностите (тип `decimal`) на покупките на клиента (по подразбиране списъкът е празен)

Нека този клас също има:

- публично достъпен уникален идентификационен код `ID` на всеки отделен обект `Client` (текстова константа, започваща с буквата 'U', следвана от 6-цифрено число, чиито незначещи цифри са заменени с нули)
- GET и SET свойства за данната `purchases`
- Конструктор за общо ползване и конструктор за копиране
- Метод `decimal AveragePurchases()`, който използва LINQ да пресметне средната сумата на покупките в списъка `purchases`. Връща пресметнатата средна стойност на сумата. Да се предвиди случая, когато списъкът `purchases` е празен.
- метод `ToString()`, онаследен от клас `object` – връща форматиран `string`, включващ уникалния код `ID`, следван от средната сума на всички покупки на клиента, форматирана с два знака след десетичната запетая- виж примера накрая

Точки: 16

3. Направете class `Client` да имплементира `IUpgradable` и също :

- Имплементирайте метода `Upgrade` в class `Client` с явно цитиране на името на интерфейса `IUpgradable` като изпълните инстанцията на делегата, подаден като аргумент на `Upgrade` с текущата инстанция на class `Client` т.е. тази имплементация да реализира категоризирането на ранга на клиента според зададените с делегата действия.
- предефинирайте метод `Equals()`, онаследен от клас `object`, така че текущият обект е равен на друг `Client` обект, когато техните идентификационни кодове `ID` съвпадат

Точки: 10

4. Напишете class `GoldClient`, който да е `Client` и да има:

- Паричен кредит `credit` (`decimal`), който по подразбиране е нула
- свойства GET и SET за данната `credit`
- Конструктор за общо ползване, конструктор за копиране и конструктор по подразбиране
- метод `ToString()`, онаследен от клас `object` – връща форматиран `string` с текстово описание на базовата инстанция и допълнително, `credit` форматиран, както в примера, даден накрая

Точки: 10

5. Напишете class `Trader`, който представя бизнес организация (търговец). Нека class `Trader` да има списък `clients` (от тип `List`) от клиенти (от тип `Client`). Нека този клас също има:

- GET и SET свойства за данната `clients` на този клас без дълбоко копиране на обектите `Client` в списъка от клиенти

- Конструктор за общо ползване
- метод `ToString()`, наследен от клас `object` – връща `string`, съдържащ текстово описание на **всичките клиенти** в списъка `clients` на инстанцията от `class Trader` - виж примера накрая

Точки: 12

6. Напишете в `class Trader`:

- Метод `void SellProducts()`, който генерира от 10 до 30 продажби на стойност в интервала `[10.00, 100.00]` лева за всеки от клиентите в списъка `clients`. Броят на продажбите и стойността им да се избира за всеки конкретен клиент, чрез генератор на случайни числа (генерирането на числа с плаваща запетая в затворен интервал да се сведе до генериране на цели числа в интервала `[1000, 10000]`, които после да се разделят на `100.00`) (3 точки)
- GET и SET индексатор за достъп до клиентите в списъка `clients` на текущата инстанция по идентификационния код на съответния клиент. Индексаторът да се реализира с LINQ. Например, ако `trader` е обект от `Trader`, то `trader["U00005"]` връща обект `Client` от `clients` с идентификационния код равен на `"U00005"`. Ако няма обект `Client` от `clients` с идентификационен код `"U00005"`, индексаторът връща `null`. SET свойството на индексатора, добавя новия `Client` обект към `clients` като изтрива обекта от `clients` с индексирания идентификационен код, ако този обект съществува.

Точки: 18

В. Създайте нов проект от тип конзолно приложение, който реферира библиотеката от класове, създадена в т. А. Напишете следното в новия проект: (30 точки)

7. Напишете разширяващ метод `void UpgradeRules()` за `class Trader` който взима единствен аргумент `client` от тип `IUpgradable` и прилага следното бизнес правило за категоризиране на клиентите по ранг:

- Когато съдържанието на аргумента `client` е `GoldClient`, то към неговия кредит `credit` се добавят 3% от общата сума на направените от него покупки
- Когато съдържанието на аргумента `client` е `Client` и сумата от средната стойност на неговите покупки е по-голяма или равна на 55.00 лв, то неговото място в списъка с `clients` на текущата инстанция се заменя с обект от тип `GoldClient` (да се използва индексатора на клас `Trader`) т.е. рангът на този клиент се повишава до `GoldClient`. Новосъздаденият `GoldClient` обект да има същия списък на покупки като `client` и кредит, равен на 0 лв..

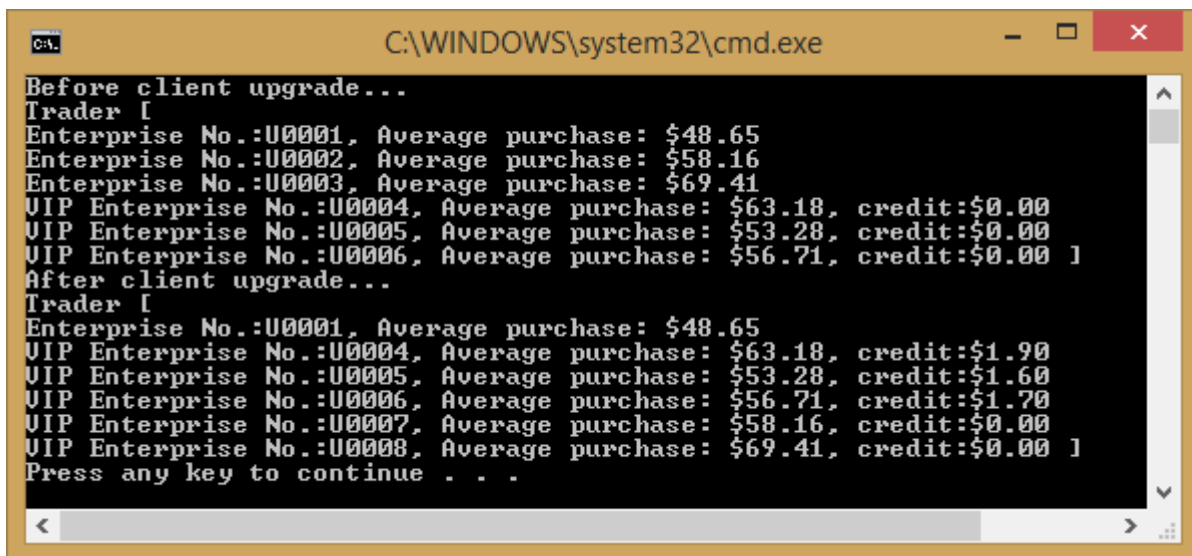
Точки: 15

8. Напишете `class PaymentTest`, в чиито `Main()` метод тествайте приложението по следния начин:

- а) Създайте списък `clients` (от тип `List`) от 3 клиенти от тип `Client` по подразбиране
- б) Добавете към списъка `clients` още 3 клиенти от тип `GoldClient` по подразбиране

- c) Създайте обект *Trader* тип *Trader* , който има списък *clients* от клиенти.
- d) Генерирайте продажби за клиентите на *Trader* с метода му *SellProducts*
- e) Приложете правилата за категоризиране по ранг на клиентите в списъка на *Trader*, реализирани в разширяващия метод *UpgradeRules*. Изпълнете метод *UpgradeRules* като аргумент на съответния метод *Upgrade* на клиентите в списъка на *Trader*
- f) Изведете на стандартен изход текстово описание на клиентите на *Trader* след извършената категоризация по ранг на клиентите - виж примера накрая

Точки:15



```
C:\WINDOWS\system32\cmd.exe
Before client upgrade...
Trader [
Enterprise No.:U00001, Average purchase: $48.65
Enterprise No.:U00002, Average purchase: $58.16
Enterprise No.:U00003, Average purchase: $69.41
VIP Enterprise No.:U00004, Average purchase: $63.18, credit:$0.00
VIP Enterprise No.:U00005, Average purchase: $53.28, credit:$0.00
VIP Enterprise No.:U00006, Average purchase: $56.71, credit:$0.00 ]
After client upgrade...
Trader [
Enterprise No.:U00001, Average purchase: $48.65
VIP Enterprise No.:U00004, Average purchase: $63.18, credit:$1.90
VIP Enterprise No.:U00005, Average purchase: $53.28, credit:$1.60
VIP Enterprise No.:U00006, Average purchase: $56.71, credit:$1.70
VIP Enterprise No.:U00007, Average purchase: $58.16, credit:$0.00
VIP Enterprise No.:U00008, Average purchase: $69.41, credit:$0.00 ]
Press any key to continue . . .
```