СОФИЙСКИ УНИВЕРСИТЕТ "КЛ. ОХРИДСКИ"
ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА

Писмен изпит No. 1b

Предмет: **Обектно ориентирано програмиране на C#.NET**
Преподавател: д-р. Е. Кръстев **Студент** :
Дата: **Време: 120 min**
Инструкции: Отбележете отговорите на теоретичните въпроси на този лист, а пълния набор от файлове необходими за решаване на програмата запишете на флопи диск.
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 на **C#.NET**:

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

при намиране решението на следната **задача за обслужване на поръчки за доставка на продукти, чието налично количество и доставка се управлява от отдалечен обект на сървера на фирмата доставчик на продуктите.** Отделните клиенти правят поръчки в **WPF графични** прозорци на ***MS Windows*** приложения, които позволяват да се избере продукт и да се поръча количество избрания продукт, чрез заявка към **уеб услуга** на фирма доставчик. Фирмата доставчик следи за наличните количества и актуализира наличните количества, когато те паднат под определени прагови стойности. Поръчките на клиентите се **записват в текстов файл на уеб сървера** на фирмата доставчик.

Решете поставената задача в описаната по- долу **последователност**:

А. Създайте проект *TradeServices* на C#.NET от тип *Class Library* със следното съдържание

1. **Дефинирайте в проекта *TradeServices* изброим тип *Category*, който съдържа следните константи** `SOFTWARE, HADRDWARE, EBOOKS`

   3 точки

2. **Добавете *class Product* в проекта *TradeServices* . Нека *Product*** обектите да **имат свой публично** достъпен (**уникален**) **пореден *string*** номер (***string* константа** за съответния обект), който започва с "***P-***", следвано от **четирицифрено число**, чиито **незначещи цифри са запълнени с нули**. Нека този номер се **представя** от **data member** именуван ***ID*** и се **инициализира** в конструкторите на class ***Product***.
   Нека *Product* обекта **има още:**
   - **данна** `productCategory` **от тип *Category***
   - **данна** `qty- int` **стойност на наличното количество на продукта**

- **данна** `reorderLevel int` **стойност на минималното количество на продукта, при което се прави нова доставка**

Напишете:

- ***set и get*** **свойства** за всички `private` данни
- **конструктор** за общо ползване и **конструктор за** копиране
- **предефинирайте** метода ***ToString(),*** който да **извежда** `ID` и **всички останали данни на обекта**, всички подходящо **форматирани**.

12 точки

3. **Добавете в проекта *TradeServices* дефиницията на WPF *UserControl*** с име ***OrderProduct***, който **има:**
   - **Етикет *Product*** и **ComboBox** `cboProduct` за **избиране на код** от падащ списък с кодове на продукти
   - **Етикет *Quantity*** и **текстово поле** `txtQty` за въвеждане **количество** за поръчка от избрания продукт (**по подразбиране** дефинирайте **0**)
   - **Бутон *Order*** и **бутон *Cancel***

   Дефинирайте `get` **свойство** за **ComboBox** `cboProduct`, което да връща референция към този обект. **При натискане** на бутона ***Cancel*** **задайте 0 в текстовото поле** `txtQty`.

   8 точки

4. **Дефинирайте в проекта *TradeServices delegate OrderHandler*** и **клас *OrderEventArgs* с аргументи** на събитие ***Order*** . **Дефинирайте** това събитие в ***OrderProduct*** и го **създайте** (fire) **при натискане** на бутона ***Order,*** за **да предостави за обработка** на потребителя **избрания код на продукт** и **количеството от този продукт**, въведени в `cboProduct` и `txtQty` на създадения ***UserControl.*** **Имената** и **структурата** на делегата и класа с аргументите да са в **съответствие с общоприетия стил за програмирането им**

   15 точки

5. **Напишете нов проект(*към същия Solution*), WCF приложение *TradeSOAPService,* което дефинира SOAP уеб услуга. Нека тази уеб услуга се дефинира с интерфейс *IOrderWService*, който има *:***
   - Метод `Retrieve` **,** който **не взима аргументи** и **връща масив** от елементи **от тип** ***Product*** ***(служи за извличане на наличните продукти за извършване на поръчки*)**
   - Метод `Update`**, който взима аргументи** `string sender, string productID, int qty` **и не връща данни.** ***(служи за записване в текстов файл на поръчка от*** `sender` ***за*** `productID` ***и поръчаното количество*** `qty` **)**

   6 точки

6. **Нека *TradeProducts* е класът, който имплементира интерфейса *IOrderWService* на уеб услугата. Маркирайте този клас с анотацията** `[ServiceBehavior(InstanceContextMode = InstanceContextMode.Single)]` **Добавете в този клас**
   - **данна** `products` **от тип** `Dictionary<string, Product>` **таблица** от ***код на продукт*** **и продукта, съответстващ на** кода

   Инициализирайте `products` **в конструктора по подразбиране на *class TradeProducts* като изберете с генератор на случайни числа:**
   - **Броят на** `products` **в интервала** `[6, 11]`
   - **Категорията** `Category` **на всеки продукт**
   - **Праг за доставка на всеки продукт в интервала** `[6, 12]`
   - **Приемете за наличното количество на всеки продукт 12 бр.**

   8 точки

7. **Нека <u>конструктора по подразбиране</u> на *class TradeProducts* стартира нишка**, която в **безкраен цикъл** да **проверява** и **актуализира `thread-safe` стойността на наличното количество на всеки от елементите** на `products`, чието **налично количество** е **под прага за доставка**. **Актуализацията да се извършва** като към наличното количество **се добавя случайно генерирано** число в интервала **`[preorderLevel, preorderLevel + 11]`**
   В края на всяка итерация от цикъла. Нишката да „заспива" за 1000 **ms**

   10 точки

8. **Нека *class TradeProducts* имплементира интерфейс *IOrderWService*, при което:**
   - **методът** `Retrieve()` връща **`thread- safe` стойности** на `Dictionary<string, Product>` данната `products` като **масив от елементи** `Product` (4 точки)
   - **методът** `Update()` **отваря** и **добавя `thread- safe`** към **текстов файл** `reorder.txt` **запис** състоящ се от **име на клиент** (първия **аргумент** на `Update()`), **описание на заявен продукт** (стойност на `products` **с ключ** втория **аргумент** на `Update()`) и заявено количество от продукта (третия **аргумент** на `Update()`) (8 точки)

     12 точки

9. **Напишете нов проект (*към същия Solution*), WPF приложение** `ProductClientApp`. Нека този проект съдържа **дефиницията на *class OrderClient***, който е **WPF форма** (**прозорец**) и **служи за поръчки** на стоки със съответните количества.
   Добавете в тази форма **WPF *UserControl*- а** с име ***OrderProduct* , създаден в предходния проект**, като **<u>визуална контрола</u>** и задайте име ***orderProduct*** на инстанцията на тази контрола **след добавянето** й в **WPF формата.**
   Нека при **<u>активиране</u> WPF формата** да приема **заглавие**, което да се образува от текста **„*Order client*"** и **случайно генерирано число** в интервала ***[1, 1000]*** `(виж примера в края на текста)`

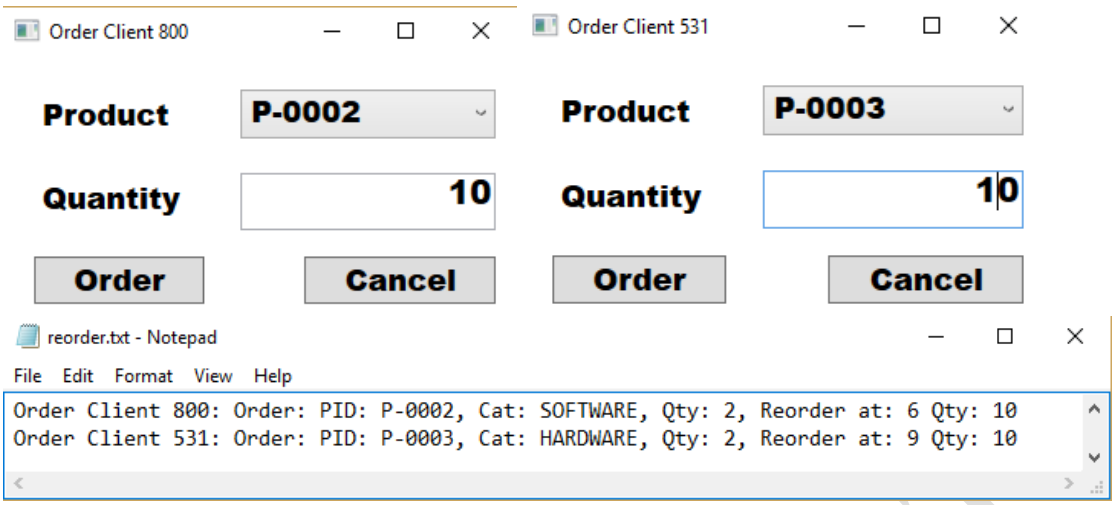
   10 точки

10. Добавете в *class OrderClient* референция ***client*** към уеб услугата ***IOrderWService***. В конструктора по подразбиране на *class OrderClient*
    - Инициализирайте референцията ***client***
    - Инициализирайте **`thread safe` списъка със стойности на *ComboBox*-а на *UserControl-a orderProduct* да показва *ID* кодовете на елементите *Product* от масива, получен от метода *Retieve()* на** `orderService`
      <u>Упътване</u>: Използвайте **`Dispatcher.CheckAccess()`** и **`Dispatcher.Invoke()`** с ***orderProduct*** **вместо `InvokeRequired()` и `Invoke()`**

      12 точки

11. **<u>Абонирайте</u> *orderProduct*** в *class OrderClient* **за събитието *Order на WPF* контролата. При обработката на това събитие да се изпълнява** метода ***Update()*** **на** `orderService`, където **първият аргумент** е заглавието (*Title*) на WPF формата *OrderClient*, **вторият** и **третият** аргумент са кодът на продукта и количеството от него, подадени от **обекта на събитието** *Order* на ***UserControl-a orderProduct (т.е. OrderEventArgs*** **)**

    4 точки


Order Client 800
Product
P-0002
Quantity
10
Order
Cancel
Order Client 531
Product
P-0003
Quantity
10
Order
Cancel
reorder.txt - Notepad
File Edit Format View Help
Order Client 800: Order: PID: P-0002, Cat: SOFTWARE, Qty: 2, Reorder at: 6 Qty: 10
Order Client 531: Order: PID: P-0003, Cat: HARDWARE, Qty: 2, Reorder at: 9 Qty: 10