Софийски университет „Св. Климент Охридски“
Факултет по математика и информатика

ИЗПИТ

курс Обектно-ориентирано програмиране
специалност Софтуерно инженерство
летен семестър 2018/2019 г.

Сесия 1
Времетраене: 2 часа и 30 минути

Изисквания за предаване:

- Предаване на решенията от изпита става като .zip архив със следното име: Exam_SI_(курс)_(група)_(факултетен_номер), където:
  - (курс) е цяло число, отговарящо на курс (например 1);
  - (група) е цяло число, отговарящо на групата Ви (например 1);
  - (факултетен_номер) е цяло число, отговарящо на факултетния Ви номер (например 63666);
- Архивът да съдържа само изходен код (.cpp и .h файлове) с решение отговарящо на условията на задачите, като файловете изходен код за всяка задача трябва да са разположени в папка с име (номер_на_задача);
- Качване на архива на посоченото място в Moodle;

Пример за .zip архив за изпита: Exam_SI_1_1_63666.zip

Задача 1. Матрица (40т.)

Да се реализира шаблон на клас `Matrix`, който представлява двумерен масив от елементи.

Да се реализират следните методи:

- Конструктор - заделя матрица с размер 2x2 (всички елементи са със стойност по подразбиране)
- Конструктор с параметри - размер на матрицата (всички елементи са със стойност по подразбиране)
- Конструктор за копиране
- Оператор=
- Деструктор
- `setAt(unsigned int x, unsigned int y, T element)` - променя елемента на позиция (x,y) (*x* и *y* започват от **1**)
- `getAt(unsigned int x, unsigned int y)` - връща елемент на позиция (x,y) (*x* и *y* започват от **1**)
- `transpose()` - транспонира матрицата

Примерни стойности и вход/изход:

```
Matrix<int> test1(2,2);
test1.setAt(1,1,5);
test1.setAt(1,2,3);
test1.setAt(2,1,4);
test1.setAt(2,2,1);
/*
(5,3)
(4,1)
*/
test1.transpose();
/*
(5,4)
(3,1)
*/
std::cout << test1.getAt(1,2); //4
```

Забележка: Не е разрешено да ползвате класове от STL библиотеката.

Задача 2 Магазин (40т.)

Да се реализира софтуер за магазин. В магазина ще се продават следните типове артикули - **плод** и **зеленчук**

Да се напишат следните класове и техните член-данни/методи:

Клас `StoreItem` (абстрактен):

- Член-данни:
  - Тип на продукта (enum)
  - Калории на продукта (цяло положително число) (0 по подразбиране)
  - Цена (дробно число) (0 по подразбиране)
- Методи:
  - Get() за всяка член-данна
  - Set() за всяка член-данна
  - Конструктор с параметри

Клас `Fruit`: (наследява `StoreItem`)

- Член-данни:
  - Тип на продукта: плод
  - Име на плода (string)
  - Цвят на плода (string)
- Методи:
  - getName() - връща името на плода
  - Set() за всяка член-данна
  - Конструктор с параметри
    - Оператор > - сравнява два продукта по брой калории
  - print() - да изведе информацията на продукта на конзолата

Клас `Vegetable`: (наследява `StoreItem`)

- Член-данни:
  - Тип на продукта: зеленчук
  - Име на зеленчука (string)
  - Сорт (string)
- Методи:
  - getName() - връща името на зеленчука
  - Set() за всяка член-данна
  - Конструктор с параметри
  - Оператор == - сравнява два продукта по сорт
  - print() - да изведе информацията на продукта на конзолата

Клас `Shop`:

Служи като контейнер за артикулите в магазина.

- Член-данни:
  - Хетерогенен контейнер за атрикулите в магазина
- Методи:
  - Добавяне на продукт (приема продукта като аргумент)
  - Премахване на продукт (приема индекс)
  - Промяна на цена на продукт (приема индекс и цена)
  - Промяна на име на продукт (приема индекс и ново име)

- Извежда информация за всички продукти на конзола
- Намира плода с най-много калории

Примерни стойности:

```
Shop shop;

Fruit t1;
t1.setName("Melon");
t1.setColor("Yellow");
t1.setCalorie(30);
t1.setPrice(1.50);

Vegetable t2;
t1.setName("Potato");
t1.setSort("Fresh");
t1.setCalorie(100);
t1.setPrice(0.80);

shop.add(t1);
shop.add(t2);
shop.changePrice(1, 1);
```

Забележка: Разрешено е да ползвате класове от STL библиотеката.