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

# ДОМАШНА РАБОТА №3

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

СРОК ЗА ПРЕДАВАНЕ: посочен в Moodle

## Изисквания за предаване:

- Предаване на домашното в указания срок от всеки студент като .zip архив със следното име: (номер_на_домашно)_SI_(курс)_(група)_(факултетен_номер), където:
  - (номер_на_домашно) е цяло число, отговарящо на номера на домашното за което е отнася решението (например 1);
  - (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), където (номер_на_задача) е номера на задачата към която се отнася решението;
- **Разрешено** е да ползвате класове от библиотеката STL като std::string, std::vector, std::stack и др.
- Качване на архива на посоченото място в Moodle;

Пример за .zip архив за домашно: 3_SI_1_1_63666.zip

Предложено е примерно именуване на класовете, полетата и функциите. По ваше желание може да ги промените, за да са по ясни и лесни за разбиране. При необходимост реализирайте конструктори, деструктор, getters, setters, допълнителни методи.

## 1. Реализирайте клас Point2D, който има:

- Координата **double x;**
- Координата **double y;**
- Метод **getDistanceTo**, който приема **Point2D** и пресмята разстоянието до него от текущия обект.

## 2. Реализирайте клас Point3D, който наследява Point2D и има:

- Координата **double z;**
- Метод **getDistanceTo**, който приема **Point3D** и пресмята разстоянието до него от текущия обект.

## 3. Реализирайте абстрактен клас Entity [i], който има:

- **id** - Цяло число, генерира се автоматично, всеки обект има уникално **id;**
- **name** – string;
- **location**, който е или **Point2D** или **Point3D** [ii];
- **Enum type**, който ще следи какъв е типа на наследените класове на **Entity**. *Пояснение: Или реализирайте Enum-а в класа **Entity** или в глобалния scope, но ако е в глобалния внимавайте да не го замърсите* [iii]*;*
- Метод **isAlive**, който винаги връща **true;**
- Метод **getDistanceTo2D**, който приема **Entity** и връща разстоянието между него и текущия обект като третира **location** като **Point2D;**
- Метод **getDistanceTo**, който приема **Entity** и връща разстоянието до него от текущия обект. *Пояснение: Ако и двата обекта пазят 3D **location**, то трябва да се пресметне разстоянието им като се използват и трите координати. Ако поне единият обект има 2D **location**, то трябва да се пресметне разстоянието им като се използват само **x,y** координатите им.*
- Метод **moveTo**, който приема аргумент **Point3D** или **Point2D** и променя **location** на текущия обект да е същия както на аргумента. *Пояснение: Ако аргументът и **location** в текущия обект са от един и същ тип да се копира нормално. Ако аргументът и **location** са от различен тип да се копира само информацията, която може. Ще се получи загуба на данни, но не е фатално.*
- Метод **moveTo**, който приема **Entity** и променя **location** на текущия обект да е същия като на подадения аргумент. *Пояснение: Използва същите правила като горния метод.*

## 4. Реализирайте клас Player, който наследява Entity и има:

- **Enum type = Player;**
- **damage** - цяло число;
- **health** - цяло число;
- Метод **isAlive**, който връща true, ако **health > 0;**
- Метод **attack**, който приема **Player** или **Mob** и отнема **damage** от неговия health ако дистанцията между двата обекта е по-малка от **5** [iv]**;**

## 5. Реализирайте клас NPC, който наследява Entity и има:

- **Enum type = NPC;**
- Метод **isAlive**, който винаги връща true;

## 6. Реализирайте клас Mob, който наследява Entity и има:

- **Enum type = Mob;**
- **damage** - цяло число;
- **health** - цяло число;
- Метод **isAlive**, който връща **true**, ако **health > 0;**
- Метдо **attack**, който приема **Player** и отнема **damage** от неговия **health,** ако дистанцията между двата обекта е по-малка от **5.**

## 7. Реализирайте клас Environment [v], който има:

- **entities** - един vector, който може да съдържа **Player** и **NPC** и **Mob** [vi];
- Метод **add**, който може да добавя **Player**, **NPC** и **Mob** към **entities** [vii];
- Метод **getAt**, който по подаден индекс **index** връща елемента на позиция **index** в **entities;**
- Метод **removeAt**, който приема индекс **index** и премахва елемента на позиция **index** в **entities;**
- Метод **generateEntities**, който създава няколко **Entity** от различен тип и ги добавя към **entities;**
- Метод **destroyEntities**, който изчиства всички обекти в **entities;**
- Метод **getClosestAliveEntity**, който приема **Player** и **Type** и връща **Entity** от вектора **entities**, който е най-близко до подадения **Player** и има същия тип като подадения аргумент и е **Alive.**

## 8. Реализирайте главна програма, която:

- Създава един **Player** *Player1;*
- Създава **Environment;**
- Напълва **Environment**-а с няколко **Entity** от различен тип;
- *Player1* минава и атакува всички **Mob** от **Environment**-а, като реда в който го прави е от най-близки, към най-далечни.

**Hints**

---

[i] Абстрактен клас е клас, който има поне един чисто виртуален метод. Абстрактният не може да има преки екземпляри и се използва за базов за други производни класове.

[ii] Указател към Point2D (да използва полиморфизъм) или може да се реализира в отделен клас Location, който има указател към Point2D

[iii] https://stackoverflow.com/questions/2503807/declaring-an-enum-within-a-class

[iv] Може да бъдат реализирани няколко метода или само един - в зависимост от това как сте реализирали останалата част от задачата

[v] Екстра (НЕ носи точки): реализирайте Environment като singleton

[vi] vector<Entity*> за да е полиморфен контейнер

[vii] Може да бъдат реализирани няколко метода или само един - в зависимост от това как сте реализирали останалата част от задачата