Задача 1. Сортиране на стек от цели числа

- Реализирайте клас IntStack (разширяващ се стек или свързан стек) от цели числа;
- Реализирайте член-функция за достъп print, който извежда елементите на стека от върха до дъното на стека;
- Напишете външна функция sortStack, която приема като аргумент стек от цели числа и връща стек, чиито елементи са сортирани по големина като на върха на стека е най-големия елемент.
- Напишете главна програма, която:
    - чете елементите на стека и ги вмъква в стека;
    - извежда елементите на стека;
    - извиква sortStack;
    - извежда елементите на сортирания стек.

Задача 2. DNS Cache

Операционните системи и браузерите използват DNS Cache – временна таблица, която съдържа записи за всички последно посетени уеб сайтове и други интернет домейни.

Да се моделира DNS Cache като за целта:

- Създайте клас или структура DNSRecord, който има следните член-данни:
    - domainName – име на домейна (низ);
    - ipAddress – IP адрес на домейна (низ).
- Създайте клас DNSCache, който:
    - Поддържа динамичен разширяващ се масив от DNS записи
    - Съответни конструктори, деструктор и оператор за присвояване;
    - Метод add за добавяне на DNS запис
    - Метод lookup, който по име на домейна връща IP адреса на домейна, ако го има в кеша или NULL – ако го няма;
    - Метод flush за изчистване на кеша;
    - Метод print, който извежда всички записи в кеша.
- Напишете главна програма за тестване на класа.

Задача 3. Онлайн магазин

Да се моделира онлайн магазин (Store) за определени продукти (тениски, ризи, маратонки, GSM, лаптопи, автомобилни гуми, и др.). За целта:

- Изберете продукт, за който ще реализирате магазина и определете характеристиките на продукта (например марка, модел, категория, цвят, размер, операционна система, памет, и др.) Да се дефинира клас Product, който има следните член-данни:
    - SKU (stock keeping unit) – уникален номер на продукта (задължително);
    - Brand – марка (задължително)
    - Model – модел (задължително)
    - Category – категория (ако е приложимо)
    - Price – единична цена (задължително)
    - Count – брой налични продукта в магазина (задължително)
    - Други член данни в зависимост от избрания продукт и определените характеристики.
    - Използвайте коментари за класа и член-данните, за да е ясно какъв е избраният продукт и какви са неговите характеристики.
- Да се дефинират съответни методи за създаване, четене, запис и промяна на продукти;
- Да се дефинира клас Store, който съдържа динамичен списък (масив) от продукти в магазина, заедно с техния брой. Да се дефинират съответни методи за:
    - Добавяне на нов продукт;
    - Изтриване на продукт;
    - Промяна на продукт;
    - Извеждане на екрана на списък с наличните продукти и техния брой;
- Да се дефинира главна програма, реализираща следното меню:
  A Add new product
  X Delete product
  C Change product
  D Display products
  Q Quit

Задача 4. MagicBox

Магьосникът Маг-О-Кодерски е магьосник в местното за ФМИ заведение BeerOverflow. Там той изпълнява всяка вечер своя коронен номер - всеки посетител на BeerOverflow пуска нещо в "магическата кутия" на Маг-О-Кодерски, а в края на вечерта кутия изхвърля един предмет във въздуха, на случаен принцип.

Маг-О-Кодерски също е и програмист. Той иска да напише програма, в която да въвежда променливи от даден тип (един път може да са int, друг път char, а трети път - Rational), и при извикване на даден метод, "кутията" да му връща случайна променлива от вече въведените.

Да се напише клас MagicBox със следните член-данни:

- Динамичен масив (типа на променливите не е ясен предварително, но знаем че всички променливи са от един тип)

Методи:

- insert – добавя елемент към кутията;
- pop – премахва случаен елемент от кутията. Ако кутията е празна, да показва подходящо съобщение;
- list – извежда на екрана списък с елементите в кутията.

Може да дефинирате допълнителни методи и член-данни.

Задача 5. MMORPG игра
Реализирайте следните класове, които ще служат като част от MMORPG игра

Клас GameCharacter:

GameCharacter е базовият герой - той не може да бъде избран от играч, само служи като основа на другите класове

Член-данни:
- Име на героя
- Точки живот (HP)
- Точки за магия (MP)
- Точки атака (AP)
- Точки защита (DP)

Методи:
- getattacked() - героят губи няколко от своите точки живот (точки атака на противника - точки защита на героя. Ако разликата е < 0, нищо не се променя в точките живот) (getattacked() приема като аргумент int - точките атака на противника).

Всички точки на героя са зададени в конструктора, и не могат да бъдат променяни

Клас Warrior:

Има следните характеристики:
- Име на героя - "Warrior"
- HP = 20
- MP = 5
- AP = 4
- DP = 1

Клас Guardian:

Има следните характеристики:
- Име на героя - "Guardian"
- HP = 40
- MP = 25
- AP = 1
- DP = 3

Клас Wizzard:

Има следните характеристики:
- Име на героя - "Wizzard"
- HP = 30
- MP = 30
- AP = 2
- DP = 2

Задача 6. Логин система

За реализирането на задачата трябва да се използва наследяване.
Реализирайте главна програма с тестове.
Реализирайте част от логин система за форум, която има следните типове потребители:

- Guest, който има следната информация
  - IP Address - IP адрес на потребителя
- User
  - User има всичката информация/функционалност от Guest и
    - Username
    - Password
    - Title
  - Password да се съхранява в криптиран формат о User не може да променя своя Title
  - User не може да променя своя Username
  - User може да променя паролата си, като трябва да даде старата си парола при смяната
- Power User
  - Power User има всичката информация/функционалност от User и
    - Reputation - колко е допринесъл потребителят за форума
  - Power User не може да променя репутацията си
  - Power User не може да променя своят Title
  - Всички други потребители освен текущия могат да дават по 1 Reputation, т.е потребител може да даде 1 Reputation на текущия потребител само ако имат различен Username
- VIP
  - VIP има всичката информация/функционалност на User
  - VIP може сам да променя Title-а си
- Admin
  - Admin има всичката информация/функционалност на VIP, Power User
  - Admin може да променя своя Username
  - Admin може да променя Username на друг потребител

HINTS

- IP address, Username, Password, Title - char *
- Reputation - int
- Примери за криптиране на паролата - XOR, ROR/ROL, NOT, Addition/Subtraction, Reverse bits, etc.
- Промяната на паролата е един метод, на който се подава стара и нова парола
- PowerUser Reputation +1 - на метода в PowerUser се подава кой дава +1 репутация (друг потребител)
- В Admin на метода за промяна на Username на потребител се подава User и нов username

Задача 7

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;
   - Метод 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 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