ООП – Практикум: Домашна работа №4

Задача 4: Ю-Ги-О!

Сето Кайба след време решава да промени своите изисквания и те вече са следните: Рефакторирайте кода, който сте използвали за решаването на предишното задание с Ю-Ги-О! (Домашна работа №3).

Реализирате 4 класа Ю-Ги-О! карти:

- **"Карта" (Card) – *абстрактен клас***
  - Име (низ)
  - Ефект (низ)
  - Рядкост/Rarity(неотрицателно целочислено число)
  - ❖ Предефинирайте operator> и operator<, като картите се сравняват по тяхното rarity.
- **"Карта чудовище" (MonsterCard):**
  - Име (низ)
  - Ефект (низ)
  - Рядкост/Rarity(неотрицателно целочислено число)
  - Атакуващи точки (неотрицателно целочислено число)
  - Защитни точки (неотрицателно целочислено число)
- **"Магическа карта" (MagicCard):**
  - Име (низ)
  - Ефект (низ)
  - Рядкост/Rarity(неотрицателно целочислено число)
  - Тип (може да е един от следните 3: "trap", "buff", "spell")
- **"Пендулум карта" (PendulumCard) - представлява комбинация от "Карта чудовище" и "Магическа карта":**
  - Име (низ)
  - Ефект (низ)
  - Рядкост/Rarity(неотрицателно целочислено число)
  - Атакуващи точки (неотрицателно целочислено число)
  - Защитни точки (неотрицателно целочислено число)
  - Тип (може да е един от следните 3: "trap", "buff", "spell")
  - Пендулум скала (*число* $\in [\ 1\ ;\ 13\ ]$)

ООП – Практикум: Домашна работа №4

Реализирайте клас тесте (Deck). Тестето трябва да:

1. Има име (низ);
2. Хетерогенен контейнер за картите
3. Има методи, които да връщат като резултат броя на:
   - картите чудовища в тестето;
   - магическитете карти в тестето;
   - пендулум картите в тестето;
   - всички карти в тестето;
4. Има метод, който добавя карта към тестето
5. Има метод, който променя карта към тестето, като за параметри приема индекс и нова карта. (Пояснение: може да се променя само карта от същия тип)
6. Има метод, който премахва съдържанието на тестето и тестето остава празно.
7. Има метод, който разбърква съдържанието на тестето(shuffle).

Реализирайте клас дуелист (Duelist). Той трябва да има:

1. Име (низ)
2. Тесте (Deck)
3. Метод, който дава достъп до тестето.
4. Булев метод, който по подаден аргумент *име на файл* записва съдържанието на тестето във файл. Връща *true*, ако операцията е успешна и *false*, ако не е.
5. Булев метод, който по подаден артумент *име на файл* чете съдържанието му и обновява тестето както е записано във файла. Връща *true*, ако операцията е успешна и *false,* ако не е.
6. Метод duel, който симулира дуел между двама дуелисти:
   - Дуел може да се състои между дуелисти, чиито тестета имат равен брой карти, иначе дуелът не може да се проведе.
   - Преди да започне, тестетата на дуелистите трябва да се разбъркат.
   - Тестетата се сравняват карта по карта, посредством предефинираните оператори > и <.
   - Когато карта от тестето на първия дуелист е > от карта от тестето на втория, се брои като точка за първия, а когато карта от тестето на първия е < от карта от тестето на втория, се брои като точка за втория
   - Ако имат еднакво rarity, не се отчита точка за нито един от дуелистите.
   - Дуелът се печели от играчът с повече точки.Ако точките им са равни, дуелът завършва с равенство.

ООП – Практикум:
Домашна работа №4

Файлът да има следния формат:

- Заглавен ред, който съдържа метаданни за името на тестето, броя карти, броя карти чудовища, броя магически карти и броя пендулум карти:
<*deckname*>|<*CardsCount*>|<*monsterCardsCount*>|<*magicCardsCount*>|<*pendulumCardsCount*>
- monsterCardsCount на брой редове, като всеки представлява информация за карта чудовище в следния вид:
<*name*>|<*effect*>|<*rarity*>|<*attackPoints*>|<*defencePoints*>
- magicCardsCount на брой редове, като всеки прдставлява информация за магическа карта в следния вид:
<*name*>|<*effect*>|<*rarity*> |<*type*>
- pendulumCardsCount на брой редове, като всеки представлява информация за магическа карта в следния вид:
<*name*>|<*effect*>|<*rarity*>|<*attackPoints*>|<*defencePoints*>|<*pendulumScale*>|<*type*>

Примерен файл, който представялява тесте "Magician deck" с 2 карти чудовища, 2 магически карти и 1 пендулум карта(общо 5 карти):

```
//Example magician_deck.txt

Magician Deck|5|2|2|1
Blue-Eyes White Dragon|This legendary dragon is a powerful engine of destruction.|0|3000|2500
Dark Magician|The ultimate wizard.|1|2500|2100
Swords of Revealing Light|Your opponent's monsters cannot declare an attack.|2|SPELL
Magic Cylinder|Inflict damage to your opponent equal to its ATK.|3|TRAP
Timegazer Magician|Your opponent cannot activate Trap Magic Cards|4|1200|600|8|SPELL

//end of example
```

ООП – Практикум: Домашна работа №4

```
//Examples/Example main
MonsterCard dragon("Blue-Eyes White Dragon", "This legendary dragon is a powerful engine of destruction.", 43, 3000, 2500);
MonsterCard magician("Dark Magician", "The ultimate wizard.", 23, 2500, 2100);
MagicCard swords("Swords of Revealing Light", "Your opponent's monsters cannot declare an attack.", 123, CardType::SPELL);
MagicCard cylinder("Magic Cylinder", "Inflict damage to your opponent equal to its ATK.", 9, CardType::TRAP);
PendulumCard timegazer("Timegazer Magician", "Your opponent cannot activate Trap Magic Cards", 3, 1200, 600, 8, CardType::SPELL);
Duelist firstDuelist("Ivan Ivanov");
firstDuelist.getDeck().setDeckname("Magician Deck");

firstDuelist.getDeck().addCard(&dragon);
firstDuelist.getDeck().addCard(&swords);
firstDuelist.getDeck().addCard(&magician);
firstDuelist.getDeck().addCard(&cylinder);
firstDuelist.getDeck().addCard(&timegazer);

firstDuelist.getDeck().shuffle();

firstDuelist.safeDeck("magician_deck.txt");

MagicCard box("Mystic Box", "Destroy one monster.", 0, CardType::SPELL);
firstDuelist.getDeck().setCard(1, &box);

Duelist secondDuelist("Gosho Goshev");
secondDuelist.getDeck().addCard(&dragon);
secondDuelist.getDeck().addCard(&swords);
secondDuelist.getDeck().addCard(&magician);
secondDuelist.getDeck().addCard(&cylinder);
secondDuelist.getDeck().addCard(&timegazer);

//End of Examples/Example main
```


ООП – Практикум: Домашна работа №4

Пояснение:

- Опитайте се да напишете максимално ефективен код, както по отношение на брой редове, така и по отношение на време за изпълнение. Помислете къде може да се намали броят на повторенията на циклите или да се намали броят на променливите, които използвате за решаване на задачата. Реализирайте задачите спазвайки добрите ООП практики, за които се говори по време на лекции и упражнения.
- Файловете с решенията може да съдържат само стандартните символи с кодове от 0-127 (не се разрешава използване на кирилица, например в стринговете или коментарите!).
- Предадените от вас решения трябва да могат да се компилират успешно на Visual C++ или GCC.
- **Разрешено е да ползвате класове от библиотеката STL като std::string, std::vector, std::stack и др**
- **Имплементирайте класовете, посочени в задачата, като спазвате принципите за Абстракция, Енкапсулация и Наследяване.**
- *Първото нещо във всеки от файловете, които предавате, трябва да бъде коментарен блок, който носи информация за съдържанието на файла. Този коментар трябва да изглежда точно така, както е показано по-долу, като в него попълните информация за Вас. За улеснение, просто копирайте дадения по-долу блок и попълнете в него необходимите данни, вместо текста, маркиран с ъглови скоби. Обърнете внимание, че на първия ред след наклонената черта има две звезди и че във файловете не може да се съдържат символи на кирилица.*

```
/**
* Solution to homework assignment 1
* Object Oriented Programming Course
* Faculty of Mathematics and Informatics of Sofia University
* Summer semester 2020/2021
*
* @author <вашето име>
* @idnumber <вашият факултетен номер>
* @task <номер на задача>
* @compiler <използван компилатор - GCC или VC>
*/
```


ООП – Практикум:
Домашна работа №4

Например един попълнен блок за студент с име Иван Иванов, Ф.н:12345, който предава задача 2, компилирана с GCC, трябва да изглежда така:

```
/**
* Solution to homework assignment 4
* Object Oriented Programming Course
* Faculty of Mathematics and Informatics of Sofia University
* Summer semester 2020/2021
*
* @author Ivan Ivanov
* @idnumber 12345
* @task 4
* @compiler GCC
*/
```

Изисквания за предаване:

- Всички задачи ще бъдат проверени автоматично за преписване. Файловете с голямо съвпадение ще бъдат проверени ръчно и при установено плагиатство ще бъдат анулирани.
- 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 12345);

    Пример за .zip архив за домашно: 4_SI_1_1_12345.zip
- Архивът да съдържа само изходен код (.cpp и .h/.hpp 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;
- Качване на архива на посоченото място в Moodle;