



# ООП – Практикум: Домашна работа №3

## Задача 1: 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 №1).

### Реализиране 3 класа Ю-Ги-О! карти (Card):

- "Карта чудовище" (MonsterCard):
  - Име (низ)
  - Ефект (низ)
  - Атакуващи точки (неотрицателно целочислено число)
  - Защитни точки (неотрицателно целочислено число)
- "Магическа карта" (MagicCard):
  - Име (низ)
  - Ефект (низ)
  - 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)
  - Атакуващи точки (неотрицателно целочислено число)
  - Защитни точки (неотрицателно целочислено число)
  - Тип (може да е един от следните 3: "trap", "buff", "spell")
  - Пендулум скала (  $число \in [1; 13]$  )

### Реализиране клас тесте (Deck). Тестето трябва да:

1. Има име (низ);
2. Съхранява неограничен брой карти чудовища;
3. Съхранява неограничен брой магически карти;
4. Съхранява неограничен брой пендулум карти;
5. Има метод, който да връща като резултат броя на картите чудовища в тестето;
6. Има метод, който да връща като резултат броя на магическите карти в тестето;
7. Има метод, който да връща като резултат броя на пендулум картите в тестето;
8. Има метод, който добавя карта чудовище към тестето;
9. Има метод, който добавя магическа карта към тестето;
10. Има метод, който добавя пендулум карта към тестето;
11. Има метод, който променя карта чудовище към тестето, като за параметри приема индекс и нова карта чудовище.
12. Има метод, който променя магическа карта от тестето, като за параметри приема индекс и нова магическа карта.





## ООП – Практикум: Домашна работа №3

13. Има метод, който променя пендулум карта от тестето, като за параметри приема индекс и нова пендулум карта.
14. Има метод, който премахва съдържанието на тестето и тестето остава празно.

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 аргумент *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.

Файлът да има следния формат:

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

Примерен файл, който представлява тесте "Magician deck" с 2 карти чудовища, 2 магически карти и 1 пендулум карта:

---

//Example magician\_deck.txt

Magician Deck|2|2|1

Blue-Eyes White Dragon|This legendary dragon is a powerful engine of destruction.|3000|2500

Dark Magician|The ultimate wizard.|2500|2100

Swords of Revealing Light|Your opponent's monsters cannot declare an attack.|SPELL

Magic Cylinder|Inflict damage to your opponent equal to its ATK.|TRAP

Timegazer Magician|Your opponent cannot activate Trap Magic Cards|1200|600|8|SPELL

//end of example

---





# ООП – Практикум: Домашна работа №3

---

//Examples/Example main

MonsterCard dragon("Blue-Eyes White Dragon", "This legendary dragon is a powerful engine of destruction.", 3000, 2500);

MonsterCard magician("Dark Magician", "The ultimate wizard.", 2500, 2100);

MagicCard swords("Swords of Revealing Light", "Your opponent's monsters cannot declare an attack.", CardType::SPELL);

MagicCard cylinder("Magic Cylinder", "Inflict damage to your opponent equal to its ATK.", CardType::TRAP);

PendulumCard timegazer("Timegazer Magician", "Your opponent cannot activate Trap Magic Cards", 1200, 600, 8);

Duelist firstDuelist("Ivan Ivanov");

firstDuelist.getDeck().setDeckname("Magician Deck");

firstDuelist.getDeck().addMonsterCard(dragon);

firstDuelist.getDeck().addMagicCard(swords);

firstDuelist.getDeck().addMonsterCard(magician);

firstDuelist.getDeck().addMagicCard(cylinder);

firstDuelist.getDeck().addPendulumCard(timegazer);

firstDuelist.display();

firstDuelist.saveDeck("magician\_deck.txt");

firstDuelist.loadDeck("magician\_deck.txt");

firstDuelist.display();

MagicCard box("Mystic Box", "Destroy one monster.", CardType::SPELL);

firstDuelist.getDeck().setMagicCard(1, box);

---

//End of Examples/Example main





# ООП – Практикум: Домашна работа №3

## 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 №3

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 1
 * Object Oriented Programming Course
 * Faculty of Mathematics and Informatics of Sofia University
 * Summer semester 2020/2021
 *
 * @author Ivan Ivanov
 * @idnumber 12345
 * @task 3
 * @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: 3\_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;

Софийски университет "Св. Климент Охридски"

Факултет по математика и информатика

