Увод

в обектно ориентираното програмиране

ООП – Основи

Съдържание

- Какво е ООП
- Основни принципи
- Клас
- Какво има в един клас
- Нива на достъп
- Наследяване

Какво е ООП?

Какво е то?

- Парадигма начин на структуриране на кода чрез класове и обекти
- ООП моделира обектите от реалния свят и взаимоотношенията между тях

Стол

- Цвят
- Материал
- Брой крака
- Позиция
- Може да бъде местен
- Празен или зает

Защо го използваме?

- Добро структуриране на кода
- Намалява сложността на кода
- Позволява преизползване на кода
- Може да се постигне абстрактност

Основни принципи

Основни принципи

- Капсулация: знае се КАКВО може да прави компонента, а не как
- Наследяване: един обект със същите свойства като друг може да го наследи
- Полиморфизъм: позволява унифицирано извършване на действия над различни обекти

Стол

```
class Chair {

String material = “wood”;

int positionX = 5;

int positionY = 4;

Color matColor = Color.RED;

void moveChair ( int x, int y){

…

}
```

Клас

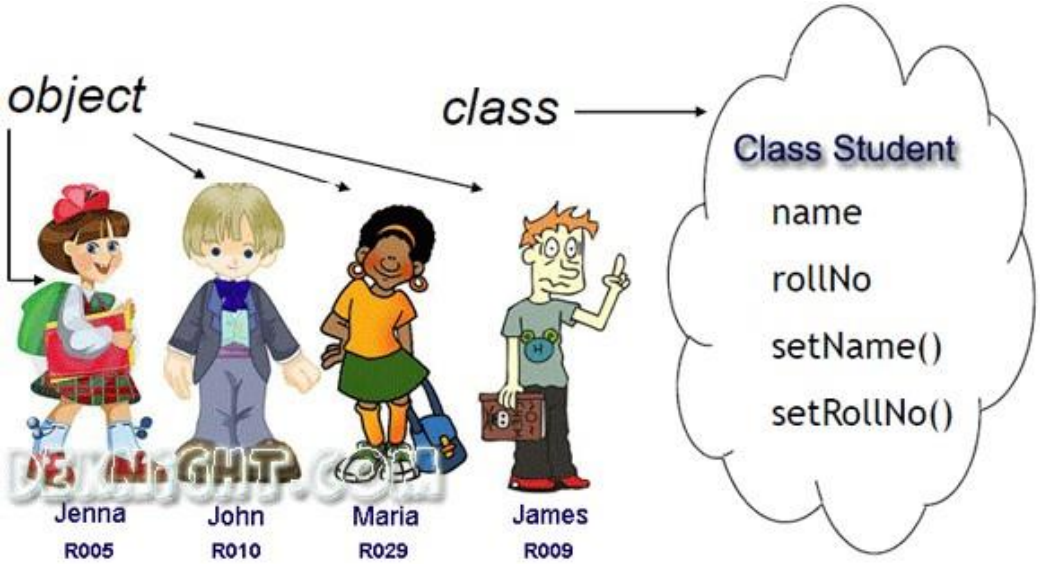
Клас

- Класът е шаблон от който създаваме обект
- Обекта е конкретен елемент от даден клас. Нарича се още инстанция на класа

```
Student pesho = new Student();

Chair zelen = new Chair();

zelen.color = Color.Green;
```



Какво има в един клас

Какво има в един клас

- Полета:
- String name;
- int age;
- Методи:
- void study(){…}
- void doHomework(){…}
- Конструктор
- Други неща

Конструктор

- Конструктора е метод, който се извиква автоматично при създаването на обект от класа и само тогава

- Използва се за да се зададат първоначални стойности на полетата и за да се направят първоначални настройки на класа

- Един клас може да има много конструктори

Задача

- Напишете клас Dog, да има поне 4 полета и поне 2 метода. Да има и конструктор.

Задача

- Напишете клас Circle с поле радиус и методи getPerimeter и getArea
- Напишете клас Square, да е подобен на Circle

Нива на достъп

Нива на достъп

Modifier	Class	Package	Subclass	World
public	✓	✓	✓	✓
protected	✓	✓	✓	✗
no modifier	✓	✓	✗	✗
private	✓	✗	✗	✗

GETTER и SETTER

GETTER и SETTER

- Ограничават достъпа до полетата.
- Предотвратяват унищожаване на данни
- Или подмяната им (още по-лошо!!!)

```
private int age;
public getAge() {
return age;
}
public setAge(int newAge) {
age = newAge;
}
```

Оператора this

- Позволява обръщане към обекта вътре от самия обект

Все едно правим

String text = “asd”;

text.substring(5);

но от вътрешността на класа String

this.substring(5) – това ще се обърне към сегашния обект

Задача

Създайте клас MyArray, който да изпълнява същите функции като масив от цели числа, но да има метод за сортиране и за намиране на най-голямото.

Наследяване

Наследяване

Класът-дете получава всички методи и полета на класа-родител

```
class Human {
  int age;
  void getOld(){
  this.age++;
  }
}
```

```
class Student extends Human {
  int number;
  void doHW(){
  }
}
```

Задача

Дефинирайте клас Animal, който да бъде наследяван от класовете Dog и Dolphin. Всички класове да имат подходящи полета, методи и конструктори.

Решение:
https://github.com/LillyMihaylova/AcademyDemos/tree/master/ObjectsDemo

Променяйте стойностите на полетата на обектите и вижте резултата при тяхното извикване.

Домашно

Задача

Дефинирайте клас FlappyBird, който наследява MovingObject, който наследява GameObject. Дефинирайте им подходящи полета и методи.

Задача

Дадено ни е училище. В училището имаме класове. Всеки клас има множество от ученици и преподаватели. Всеки преподавател има множество от дисциплини, по които преподава. Учениците имат име и уникален номер в класа. Класовете имат уникален текстов идентификатор. Дисциплините имат име, брой уроци и брой упражнения.
Задачата е да се моделира училище с Java класове (SchoolClass наследява School. Student и Teacher наследяват SchoolClass. Discipline наследява Teacher). Трябва да декларирате класове заедно с техните полета, методи и конструктори. Дефинирайте и тестов клас, който демонстрира, че останалите класове работят коректно.

Упътване: Създайте класове **School**, **SchoolClass**, **Student**, **Teacher**, **Discipline** и в тях дефинирайте съответните им полета, както са описани в условието на задачата. Не ползвайте за име на клас думата "**Class**", защото в Java тя има специално значение. Добавете методи за отпечатване на всички полета от всеки от класовете и ги извикайте в тестовия клас.

Задача

Прочетете това:

•http://www.introprogramming.info/intro-java-book/readonline/glava11-sazdavane-i-izpolzvanena-obekti/

По избор прочетете и това:

•http://www.introprogramming.info/intro-java-book/readonline/glava14-definirane-na-klasove/

•Подсказка: ще имате да го прочетете и за следващото домашно