Lecture 20: OOP - inheritance

By Ahmed E. Khaled
Recap to the previous lecture!

- The different string-related functions.
Lecture 20: OOP - Inheritance

• More about OOP concepts (Inheritance).
Inheritance

• Different kinds of objects often have a certain amount **in common**.

• Mountain bikes and road bikes? Both share the characteristics of bicycles (e.g. speed and gear).

• Yet each also defines additional features that make them different.
• Students and employees are all persons.

• They all share the characteristics of person like name and age.

• Yet each also defines additional features that make them different:
  1. Student has a GPA and graduation date.
  2. Employee has a salary and position.
• Inheritance is to establish *is-a relationships* between classes.

• Inheritance derives more-specific categories from more-generic ones (a truck is a kind of vehicle for larger items).

• Java keyword “extends” is to create this *parent-child* relationship between two classes.
Parent Class “Super class”

Child Class “Sub class”

Person
  • Name
  • Age

Student
  • GPA
  • Grad. Date

Employee
  • Salary
  • Position
• Student class “extends” Person with new properties (GPA, Grad. Date).

• Now any student has the following properties:
  1. Name
  2. Age
  3. GPA
  4. Grad. Date
• Employee class “extends” Person with new properties (Salary, Position).

• Now any employee has the following properties:
  1. Name
  2. Age
  3. Salary
  4. Position

  As any person
  As an employee
Student inherits person properties

```java
public class person {
    String name;
    int age;
}

public class student extends person {
    double gpa;
    int GradeTime;
}
```
public static void main(String[] args) {
   
    person p1 = new person();
    p1.name = "adam";
    p1.age = 18;

    student s1 = new student();
    s1.name = "alice";
    s1.age = 16;
    s1.gpa = 3.0;
    s1.GradeTime = 2018;

}
Accessibility of variables and methods

```java
public class person {
    String name;
    int age;
    
    void print_person_info(){
        System.out.println("name "+name+" age "+age);
    }
}

public class student extends person {
    double gpa;
    int GradeTime;
    
    void print_student_info(){
        System.out.println("gpa "+gpa+" gradetime "+GradeTime);
    }
}
```
public static void main(String[] args) {

    Person p1 = new Person();
    p1.name = "adam";
    p1.age = 18;
    p1.print_person_info();

    Student s1 = new Student();
    s1.name = "alice";
    s1.age = 16;
    s1.gpa = 3.0;
    s1.GradeTime = 2018;

    s1.print_person_info();
    s1.print_student_info();
}
public class student extends person {

    double gpa;

    int GradeTime;

    void print_student_name() {
        System.out.println("The student's name is "+name);
    }

    void print_student_info() {
        print_person_info();
        System.out.println("gpa "+gpa+" gradetime "+GradeTime);
    }
}
public static void main(String[] args) {

    person p1 = new person();
    p1.name = "adam";
    p1.age = 18;
    p1.print_person_info();

    student s1 = new student();
    s1.name = "alice";
    s1.age = 16;
    s1.gpa = 3.0;
    s1.GradeTime = 2018;
    s1.print_student_name();
    s1.print_student_info();
}
More subclasses for the same superclass
public class employee extends person {
    double salary;
    String position;
    void print_employee_info() {
        print_person_info();
        System.out.println("salary "+salary+" grade time "+position);
    }
}

public class student extends person {
    double gpa;
    int GradeTime;
    void print_student_info() {
        print_person_info();
        System.out.println("gpa "+gpa+" grade time "+GradeTime);
    }
}
public static void main(String[] args) {

    person p1 = new person();
    p1.name = "adam";
    p1.age = 18;
    p1.print_person_info();

    student s1 = new student();
    s1.name = "alice";
    s1.age = 16;
    s1.gpa = 3.0;
    s1.GradeTime = 2018;
    s1.print_student_info();

    employee e1 = new employee();
    e1.name = "alice";
    e1.age = 16;
    e1.salary = 3000;
    e1.position = "team Leader";
    e1.print_employee_info();
}

Inheritance properties

• Subclass inherits only “public” members (variables – functions).

• Private members are only accessible from its own class, not from the sub-class.
public class person {
    String name;
    private int age;

    void print_person_info() {
        System.out.println("name " + name + " age " + age);
    }
}

public class student extends person {
    double gpa;

    void print_student_info() {
        print_person_info();
        System.out.println("gpa " + gpa);
    }
}
public class mainclass {

    public static void main(String[] args) {

        student s1 = new student();
        s1.name = "alice";
        s1.age = 16;
        s1.gpa = 3.0;

        s1.print_student_info();
    }
}
How to access private members?
public class person {

    String name;
    private int age;

    void set_age(int new_age)
    {
        age = new_age;
    }

    int get_age(int new_age)
    {
        return age;
    }

    void print_person_info()
    {
        System.out.println("name "+name+" age "+age);
    }
}

public static void main(String[] args) {

    student s1 = new student();
    s1.name = "alice";
    s1.set_age(16);
    s1.gpa = 3.0;

    s1.print_student_info();
}
Exam II - notes

• Topics:
  1. Methods
  2. Arrays and Loops
  3. Conditions
  4. OOP and inheritance
  5. Strings

• The exam may contains questions like:
  1. Write the output of a given program.
  2. Use String-related functions to get suitable form.
  3. Write a java method/class to solve a given problem.
OOP

- Define class and objects of this class.
- Define members of the class (variables – methods).
- Public vs. Private (Set – Get).
- Constructors (Three forms).
- Inheritance structure.