

Evidence for Implementation and Testing Unit.

Gabriela Lewandowska

E15

18 September 2017

I.T 1- Demonstrate one example of encapsulation that you have written in a program.

```
public class Tile {
    private int number;
    private ArrayList<Player> players;

    public Tile(int number) {
        this.number = number;
        this.players = new ArrayList<>();
    }

    public int getNumber(){
        return this.number;
    }

    public int ArrayList<Player> getPlayers(){
        return this.players;
    }
}
```

I.T 2 - Inheritance in a program.

```
public abstract class Tile implements CanUpdatePlayerPosition {
    int number;
    ArrayList<Player> players;

    public Tile(int number) {
        this.number = number;
        this.players = new ArrayList<>();
    }

    public int getNumber() { return number; }

    public ArrayList<Player> getPlayers() { return players; }

    public void setPlayers(ArrayList<Player> players) { this.players = players; }

    public void addPlayers(Player player) { this.players.add(player); }
}
```

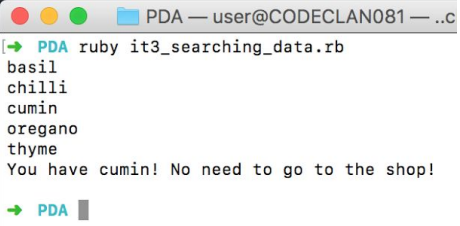
```
public class SnakeTile extends Tile {

    public SnakeTile(int number) {
        super(number);
        this.players = new ArrayList<>();
    }

    @Override
    public void updatePlayerPosition() {
        for (Player player : players) {
            player.setCurrentPosition(this.number - 5);
        }
    }
}
```

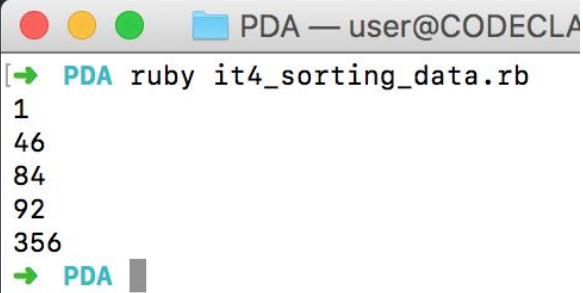
I.T 3 - Example of searching and sorting data.

```
1 cupboard = ["chilli", "thyme", "oregano", "basil", "cumin"]
2
3
4 def sort_alphabetically(array_name)
5   sorted_array = array_name.sort
6   return sorted_array
7 end
8
9 def check_if_cupboard_includes(array, needed_ingredient)
10  if array.include?(needed_ingredient)
11    puts "You have " + needed_ingredient + "! No need to go to the shop!"
12  else
13    puts "You don't have " + needed_ingredient + "! You need to buy it!"
14  end
15 end
16
17 puts sort_alphabetically(cupboard)
18
19 puts check_if_cupboard_includes(cupboard, "cumin")
20
```



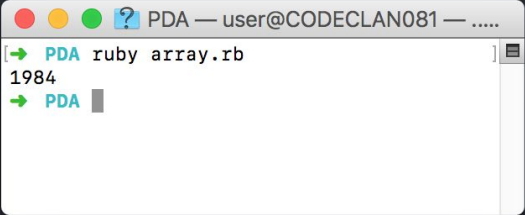
I.T 4 - Example of an array, a function that uses an array and the result

```
it4_sorting_data.rb
1 array = [1, 46, 84, 92, 356]
2
3 def sort_array(array_to_sort)
4   sorted_array = array_to_sort.sort
5   return sorted_array
6 end
7
8 puts sort_array(array)
9
```



I.T 5 - Example of a hash, a function that uses a hash and the result

```
array.rb
1 books = ["1984", "War and Peace", "Crime and Punishment", "Pride and Prejudice"]
2
3 def find_the_odd_one_out(array)
4   for book in array
5     return book if book.include?("and") == false
6   end
7 end
8
9 puts find_the_odd_one_out(books)
10
```



The terminal window shows the command `PDA ruby array.rb` being executed, which results in the output `1984`. The prompt `PDA` is visible on the next line.

I.T 7 - Demonstrate the use of polymorphism in a program

```
public class Event {
    Sport sportType;
    int maximumNumberOfCompetitors;
    ArrayList<Comparable> competitors;
    ArrayList<Comparable> rankedCompetitors;
    MedalTable medalTable;

    public Event(Sport sportType, int maximumNumberOfCompetitors) {
        this.sportType = sportType;
        this.medalTable = new MedalTable();
        this.maximumNumberOfCompetitors = maximumNumberOfCompetitors;
        this.competitors = new ArrayList<>();
        rankedCompetitors = new ArrayList<>();
    }

    public Sport getEventType() { return sportType; }

    public int getMaximumNumberOfCompetitors() { return maximumNumberOfCompetitors; }

    public ArrayList<Comparable> getCompetitors() { return competitors; }

    public MedalTable getMedalTable() { return medalTable; }

    public ArrayList<Comparable> getRankedCompetitors() { return rankedCompetitors; }

    public void addCompetitor(Competitor competitor){
        if(this.competitors.size() < this.maximumNumberOfCompetitors) {
            this.competitors.add(competitor);
        }
    }
}
```

```
public abstract class Competitor implements Comparable<Competitor> {
    private Country country;
    private int score;
    private HashMap<MedalType,Integer> medal;

    public Competitor(Country country) {
        this.country = country;
        this.score = 0;
        this.medal = new HashMap();
        this.medal.put(MedalType.GOLD, 0);
        this.medal.put(MedalType.SILVER, 0);
        this.medal.put(MedalType.BRONZE, 0);
    }
}
```



```
public class Athlete extends Competitor {  
    private String name;  
  
    public Athlete(String name, Country country) {  
        super(country);  
        this.name = name;  
    }  
  
    public String getName() { return name; }  
}
```

```
public class Team extends Competitor {  
    private ArrayList<Athlete> teamMembers;  
  
    public Team(Country country) {  
        super(country);  
        teamMembers = new ArrayList<>();  
    }  
  
    public ArrayList<Athlete> getTeamMembers() {  
        return teamMembers;  
    }  
}
```