BÁO CÁO THỰC HÀNH LAB 3 – LAB 3 REPORT

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG – Object-oriented programming

2. Working with method overloading

2.1. Overloading by differing types of parameter

A computer screen shot of a program

Description automatically generated

A screenshot of a computer program

Description automatically generated

* Adding a method addDigitalVideoDisc which allows to pass an arbitrary number of arguments for dvds:

A screen shot of a computer

Description automatically generated

Java recognizes the signature for using an array and using an arbitrary number of arguments as similar, therefore we cannot implement both methods together.

A screen shot of a computer program

Description automatically generated

In the case of adding multiple dvds to the cart, using arbitrary number of arguments should be preferrable as we do not have to create an array of dvds we want to add and can simply list out the dvds as parameters.

2.2. Overloading by differing the number of parameters

A computer screen shot of a program

Description automatically generated

A screenshot of a computer program

Description automatically generated

3. Passing parameter

A screenshot of a computer program

Description automatically generated

* For primitive data types, their values are copied to the parameters that objects receive through methods, therefore in such cases it is pass by value.
* For non-primitive data types, their references are passed on to objects instead of the actual object, so in these cases it is pass by reference. However, the references to objects are represented as their locations in the memory, and passing the references can be understood as passing the locations/addresses of objects in the memory, which will be passed by value.
* After the call of swap(jungleDVD, cinderellaDVD): references to jungleDVD and cinderellaDVD are passed to the swap method and received by the two parameters o1 and o2, i.e., o1 takes the address of jungleDVD and o2 takes the address of cinderellaDVD. Afterwards, o1 and o2 are swapped, o1 then takes the address of cinderellaDVD and o2 takes the address of jungleDVD; the two original references to the objects are not affected, so their titles remain unchanged.
* After the call of changeTitle(jungleDVD, cinderellaDVD. getTitle()): the dvd parameter takes the address of jungleDVD passed on by the method; then, dvd.setTitle(title) is called and the object referenced by dvd (i.e., jungleDVD) receives the message and changes its title; so, when we check the title of jungleDVD, it is changed.