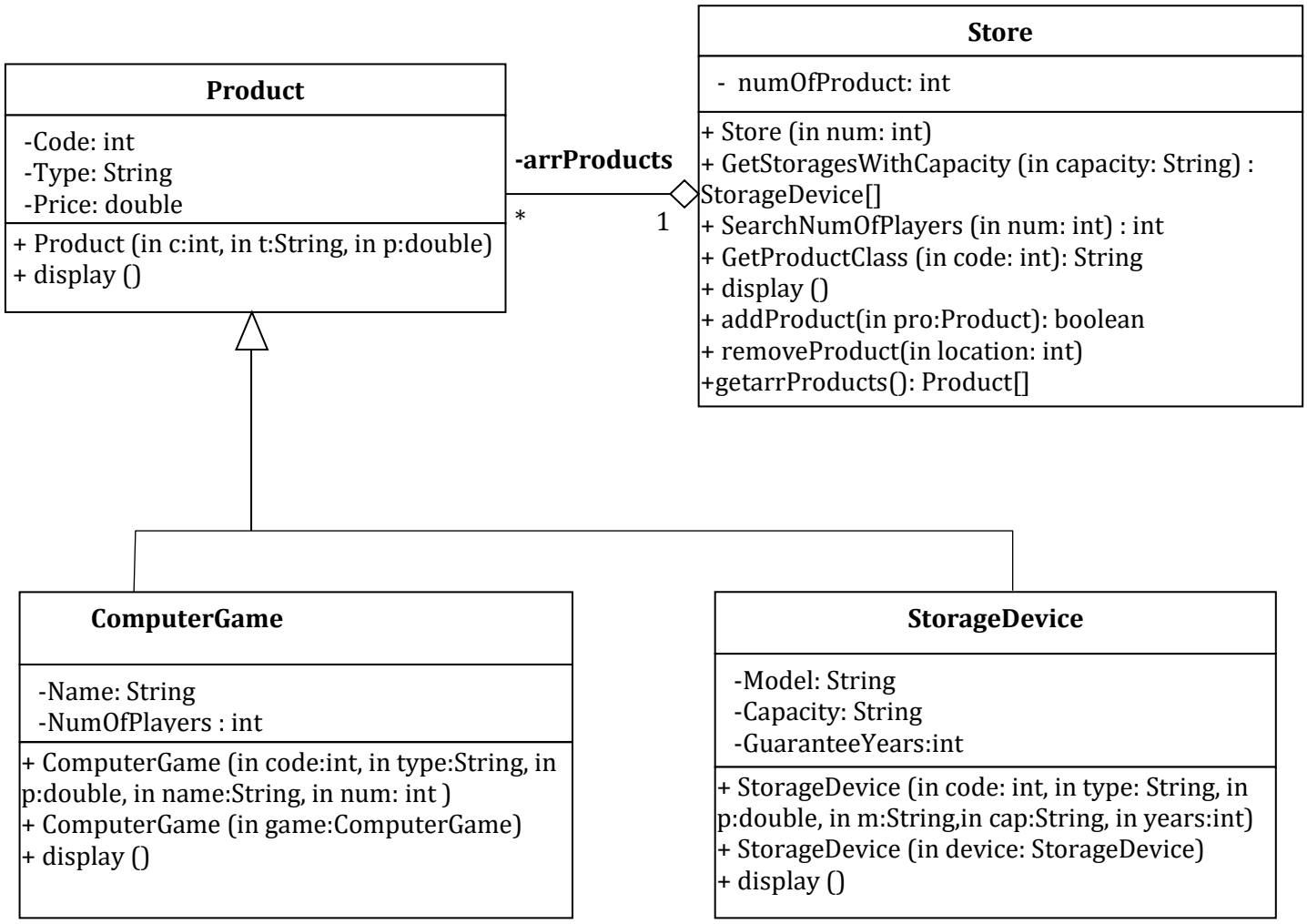
KING SAUD UNIVERSITY COLLEGE OF COMPUTER AND INFORMATION SCIENCES Computer Science Department		
CSC 113: Introduction to Programming II	**Lab_Sheet#7**	**2nd Semester 1438**

Given the following UML diagram, implement all classes with their methods using the following descriptions:



Class Product:

Code: product code
Type: type of product (ex: FlashMemory, CD,... for StorageDevice)(ex:XBOX, Sony,...for ComputerGame)
Price: the price of product
display (): Display all product information.

Class ComputerGame:

Name: Name of game

NumOfPlayers*: maximum number of players allowed*

display (): Displays all the attributes of a ComputerGame.

Class StorageDevice:

Model: the model of Storage Device

Capacity: Capacity of Storage Device (ex: 2M).

GuaranteeYears: Years of guarantee (ex: 0,1,2)

display (): Displays all the attributes of a *Storage Device.*

Class Store:

numOfProduct: *number of products currently in an object of the class Store.*

+ GetStoragesWithCapacity (in capacity: String): Receives a capacity of Storage Device and returns an array containing all Storage Device having the requested capacity.

+ SearchNumOfPlayers (in num: int): Receives number of players and returns the number of Computer Games having the specified player number. It returns 0 if no computer game has the same number of players.

+ GetProductClass (int: code): Receives a product code and returns the class name of the product. It returns null if you don't find the requested product.

+ display (): Displays the information of all products that existing in Store.

+ addProduct(in pro:Product): adds the given product at the first empty location on the array. It returns false if the array is full.

+ removeProduct(in location: int): Removes the product having the specified location, shifting the array elements such that all other products remain in the same order in the array.

+getarrProducts(): returns arrProducts array.

Write a test class having a main() method:

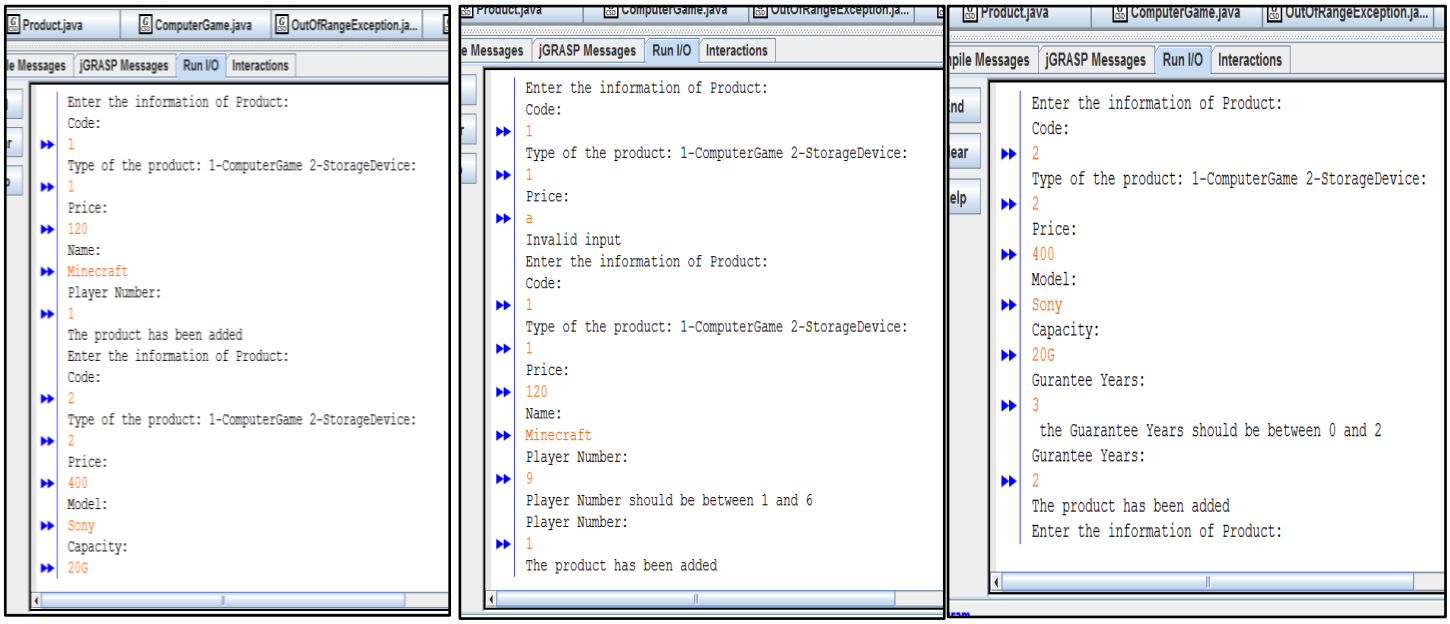
1- Create a Store object named **s** that is have 4 products.
2- Create two **StorageDevice** objects and two **ComputerGame** objects reading their data from the user.
3- Add these objects to **s** store**.**
4- Create a menu to do the following:
    a. Print storages devices that have specified capacity: ask the user to enter capacity and then retrieve and print the array containing all Storage Device having the requested capacity.
    b. Print the number of Computer Games having specified number of players: ask the user to enter the number of players and then print the retrieved number of games.
    c. Print class name of the product: ask the user enter the product code and then print the retrieved class name.
    d. Remove product having specified location: ask the user to enter the location of the product he wants remove it.
    e. Display the information of all products.

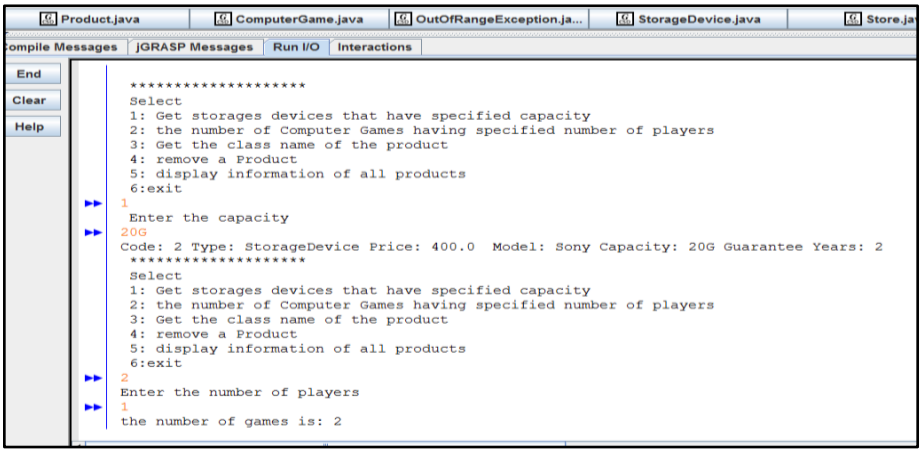
Add the following exception handling routines to your program:

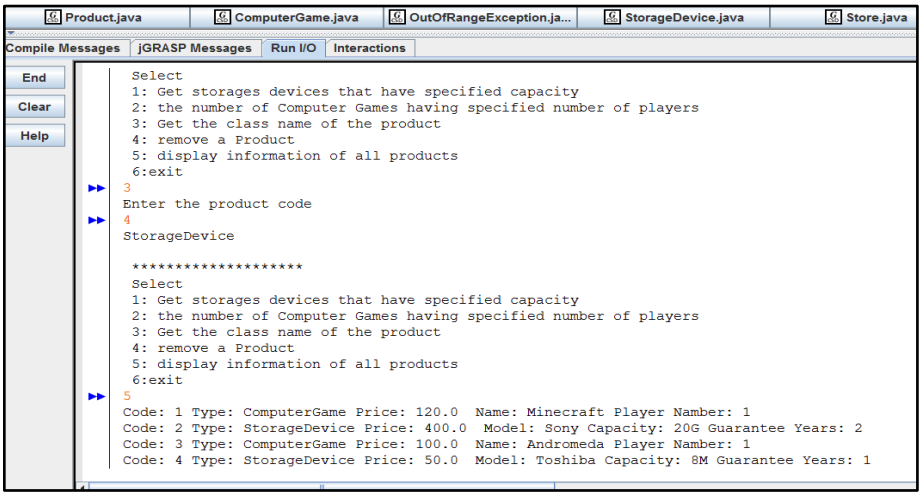
1. The constructor of the class **StorageDevice** should make sure that the *GuaranteeYears* of the storage device is within the range of [0-2]. If the *GuaranteeYears* is outside this range, a user defined exception called **OutOfRangeException** is generated and thrown to the calling environment.
2. The constructor of the class **ComputerGame** should make sure that the *NumOfPlayers* of the game is within the range of [1-6]. If the *NumOfPlayers* is outside this range, an exception of type **OutOfRangeException** is generated and thrown to the calling environment.
3. The method **SearchNumOfPlayers (int num)** will make sure that the argument ***num*** is within the range of [1-6]. If ***num*** is outside this range, an exception of type **OutOfRangeException** is generated and thrown to the calling environment, also.
4. The method **removeProduct (in location: int):**
    - Will make sure that the ***location*** is within the array bounds. Otherwise, an appropriate exception is generated. This exception should be caught in the method and the user should be allowed to enter a new index value until he enter the correct location.

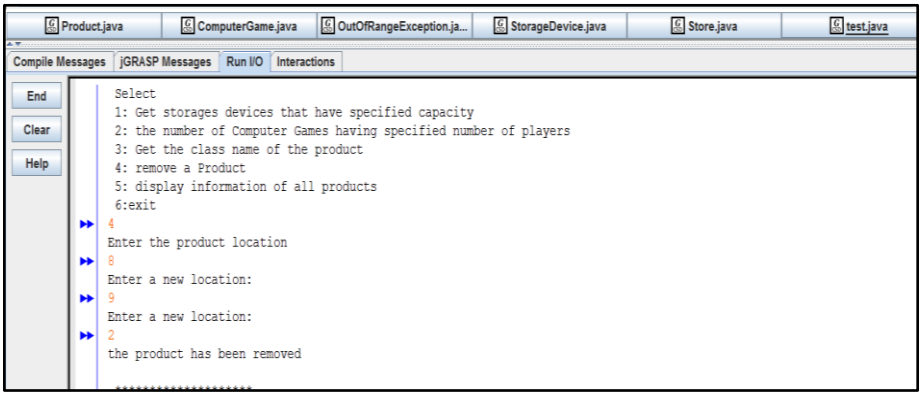
- **Hint:**
    - All exceptions thrown and not caught by the above methods should be caught in the **main()** method, such that the **main()** method allows the user to enter new values until correct the error.
    - Add any extra exceptions that may happen in the main method.

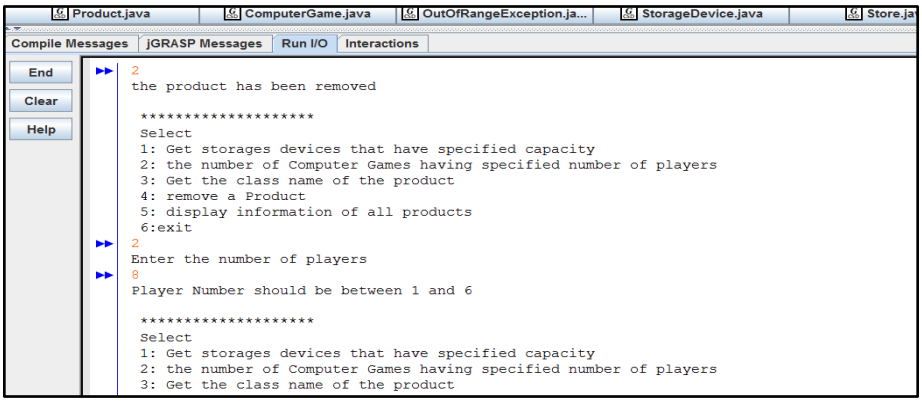
Sample Run:




Product.java
ComputerGame.java
OutOfRangeException.ja...
StorageDevice.java
Store.ja
ompile Messages
jGRASP Messages
Run I/O
Interactions
End
Clear
Help
****
Select
1: Get storages devices that have specified capacity
2: the number of Computer Games having specified number of players
3: Get the class name of the product
4: remove a Product
5: display information of all products
6:exit
1
Enter the capacity
20G
Code: 2 Type: StorageDevice Price: 400.0 Model: Sony Capacity: 20G Guarantee Years: 2
****
Select
1: Get storages devices that have specified capacity
2: the number of Computer Games having specified number of players
3: Get the class name of the product
4: remove a Product
5: display information of all products
6:exit
2
Enter the number of players
1
the number of games is: 2


Product.java
ComputerGame.java
OutOfRangeException.ja...
StorageDevice.java
Store.java
Compile Messages
jGRASP Messages
Run I/O
Interactions
End
Clear
Help
Select
1: Get storages devices that have specified capacity
2: the number of Computer Games having specified number of players
3: Get the class name of the product
4: remove a Product
5: display information of all products
6:exit
3
Enter the product code
4
StorageDevice
****
Select
1: Get storages devices that have specified capacity
2: the number of Computer Games having specified number of players
3: Get the class name of the product
4: remove a Product
5: display information of all products
6:exit
5
Code: 1 Type: ComputerGame Price: 120.0 Name: Minecraft Player Number: 1
Code: 2 Type: StorageDevice Price: 400.0 Model: Sony Capacity: 20G Guarantee Years: 2
Code: 3 Type: ComputerGame Price: 100.0 Name: Andromeda Player Number: 1
Code: 4 Type: StorageDevice Price: 50.0 Model: Toshiba Capacity: 8M Guarantee Years: 1


Product.java
ComputerGame.java
OutOfRangeException.ja...
StorageDevice.java
Store.java
test.java
Compile Messages
jGRASP Messages
Run I/O
Interactions
End
Clear
Help
Select
1: Get storages devices that have specified capacity
2: the number of Computer Games having specified number of players
3: Get the class name of the product
4: remove a Product
5: display information of all products
6:exit
4
Enter the product location
8
Enter a new location:
9
Enter a new location:
2
the product has been removed


Product.java
ComputerGame.java
OutOfRangeException.ja...
StorageDevice.java
Store.ja
Compile Messages
jGRASP Messages
Run I/O
Interactions
End
Clear
Help
2
the product has been removed
****
Select
1: Get storages devices that have specified capacity
2: the number of Computer Games having specified number of players
3: Get the class name of the product
4: remove a Product
5: display information of all products
6:exit
2
Enter the number of players
8
Player Number should be between 1 and 6
****
Select
1: Get storages devices that have specified capacity
2: the number of Computer Games having specified number of players
3: Get the class name of the product