Block 2 - Exercise for Units 1 to 4

In this exercise you are asked to implement a Java application in a source file called **Games**. There must be a public class inside this file with the same name, and inside this class you must implement the following functions:

generateNumber(int min, int max)

This function must return a random, integer number between `min` and `max` (both included). You can use `Math.random()` function for this, as we have explained in the contents of Unit 4.

generateLottery()

This function must return an array of 6 random numbers between 1 and 49 (both included). The array must be sorted in ascending order, and you must make sure that there are no repeated numbers.

checkLottery(int[] user, int[] winner)

This function must return an integer indicating how many numbers from `user` array are included in `winner` array.

playNim(int number)

This is a void function to play Nim game. In this game, we start with a set of `number` chips, and two players (user and computer in this case) alternatively substract a quantity between 1 and 3 chips from the set. The player who gets the last chip in the set loses the game.

The function must manage these turns, applying these rules:

- The user always starts playing, and it must choose in his turn an amount between 1 and 3
- The computer must choose a random number between 1 and 3 in its turn (both limits included). If there are less than 3 chips pending, then it must not exceed this limit.

You can use any additional function that you may need to complete this game.

This is an example of how it should work, assuming that we call the function to play with 20 chips:

```
Playing Nim with 20 chips.
Your turn. Choose how many chips to substract:
2
18 chips pending.
Computer substracts 1 chips.
17 chips pending
Your turn. Choose how many chips to substract:
3
14 chips pending.
Computer substracts 3 chips.
11 chips pending.
Your turn. Choose how many chips to substract:
2
9 chips pending.
Computer substracts 1 chips.
8 chips pending.
Your turn. Choose how many chips to substract:
3
5 chips pending.
Computer substracts 2 chips.
3 chips pending.
Your turn. Choose how many chips to substract:
2
1 chips pending.
YOU WIN
```

playLottery()

This is a void function to play lottery. We must ask the user to fill his 6 numbers. These numbers must be passed in a single line, separated by whitespaces (it doesn't matter if they are sorted or not, and we assume that they are valid numbers). Then, you must use `generateLottery` and `checkLottery` functions explained above to generate a random winner combination and check how many numbers are included in user's combination.

This is an example of how it should work:

```
Enter your combination:
31 3 24 23 7 10
This is the winner combination:
3 4 10 21 28 31
You have 3 hits.
```

The main function

Regarding the main function, it must receive some arguments from the command line (`String[] args` parameter):

- First parameter must be either "lottery" or "nim". If it's "lottery", then the user will play lottery game, and if it's "nim", then he will play "nim".
- If user wants to play Nim game, then there must be a second parameter, which will be an integer containing the number of chips to start playing with. This number must be converted from string to integer, and if it's not a valid number, the program must catch the corresponding exception and show the message "Incorrect number of chips".
- In any other case, the program must finish with the message "Wrong input". This last case includes these situations:
  - There are no arguments
  - First argument is other than "lottery" or "nim"
  - Second argument for "nim" game is <= 0 or > 30.

Evaluation criteria

This exercise will be evaluated as follows:

- `generateNumber` function: 1 point
- `generateLottery` function: 1.5 points
- `checkLottery` function: 1.5 points
- `playNim` function, including any other additional function that you may need: 2 points
- `playLottery` function: 1.5 points
- `main` function, including argument management: 1.5 points
- Code cleanliness (use of English, appropriate variable names, comments, spacing...): 1 point