Exercise 1:

a) Write a java program that reads an integer max and prints all power of 2 numbers between 2 and max (inclusive).

b) Having done (a), adjust the program by making sure the entered value is greater or equals to 2. If it is less, then print "Error", and ask the user for another value.

Enter an integer: 50
2.0
8.0
16.0
32.0

Exercise 2:

The factorial of a non-negative integer n is written as n! (pronounced “n factorial”) and is defined as follows:

\[ n! = 1 \text{ (for } n = 0) \]

and

\[ n! = n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot 1 \text{ (for values of } n \text{ greater than or equal to } 1) \]

For example, \( 5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \), which is 120.

Write an application that reads a nonnegative integer and computes and prints its factorial. If a negative number was typed, print "Error", and ask the user for another integer.

Enter a nonnegative integer: -2
Error
Enter a nonnegative integer: 5
120
Exercise 3:

<table>
<thead>
<tr>
<th>Student</th>
<th>TestStudents</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ id: int</td>
<td>+ main(args: String[])</td>
</tr>
<tr>
<td>+ nbCourses: int</td>
<td></td>
</tr>
<tr>
<td>+ totalPoints: int</td>
<td></td>
</tr>
<tr>
<td>+ totalCredits: int</td>
<td></td>
</tr>
<tr>
<td>+ gpa: double</td>
<td></td>
</tr>
<tr>
<td>+ money: double</td>
<td></td>
</tr>
</tbody>
</table>

a). Implement the following class that represents Student and contains the following attributes:

- **id**: student id.
- **nbCourses**: number of courses taken by student in last semester.
- **totalPoints**: total points for student in last semester.
- **totalCredits**: total number of hours taken by student in last semester.
- **gpa**: the student GPA in last semester.
- **money**: student reward.

b). Using the class Student, write the main class TestStudents that do the following:

- Ask the user to enter the number of students to process.
- For each student:
  - Read the id.
  - Read nbCourses.
  - For each course:
    - Read course credits (hours/integer).
    - Read student grade (A = 5, B = 4, C = 3, D = 2, F = 1).
    - Update/Calculate student totalCredits (totalCredits = \( \Sigma \) credits(i)).
    - Update/Calculate student totalPoints (totalPoints = \( \Sigma \) grade(i) * credits(i)).
  - Calculate gpa (gpa = totalPoints / totalCredits).
  - Calculate money (800 if gpa <= 2.0, 1500 if gpa >= 4.5, otherwise 1000).
  - Print student id, gpa and money.
- Print average GPA of all the students with totalCredits greater than 10.