CSC1003 Assignment 3

## Important Notes:

1. The assignment is an individual project, to be finished on one’s own effort.
2. The work must be submitted before 6pm Dec. 7, 2022 (Wednesday), Beijing Time. This is a firm deadline. No late submissions are accepted.
3. Plagiarism is strictly forbidden, regardless of the role in the process. Notably, ten consecutive lines of identical codes are treated as plagiarism. Depending on the seriousness of the plagiarism, 30%-100% marks will be deducted.
4. Each student is only permitted to submit code to OJ **up to FIVE times**. Only the **last submission** will be used in evaluation of assignment marks.

## Marking Criterion:

1. The full score of the assignment is 100 marks.
2. There are three java programs to be submitted. The first two programs have 25 marks each. The third program (with function A) has 50 marks. A submission obtains the full marks of each question if and only if it passes the test case respectively.
3. For the third program, an extra bonus of 20 marks will be given if the program supports both functions A and B. (However, please be noted: Even with bonus score, a student’s maximum score averaged over all the four assignments is still limited to 100 at most.)

## Running Environment:

1. The submissions will be evaluated in the course’s OJ system running Java SE version 17 and Linux platform.
2. The submission is only allowed to import four packages of (java.lang.\*; java.util.\*; java.math.\*; java.io.\*) included in Java SDK. No other packages are allowed.
3. All students will have an opportunity to test their programs in the OJ platform prior to the official submission.

## Submission Guidelines:

1. Inconsistency with or violation from the guideline leads to marks deduction.
2. It is the students’ responsibility to read this assignment document and submission guidelines carefully and in detail. No argument will be accepted on issues that have been specified clearly in these documents.

## Program I:

|  |
| --- |
|  |
| An example of input “10 2 3 1”: There are ten discs to move. At the beginning the discs are on Post 2. The objective is to move all discs to Post 3. Post 1 can be utilized during the move. |

Write a Java program (HanoiTower.java) for the Puzzle of Hanoi Tower (see course slides “CS.6.Recursion.pdf”) according to the following requirement.

1. The three posts of the puzzle are marked as “1”, “2”, and “3” respectively from left to right.
2. The number of discs, the post where the discs are put at the beginning, the target post to move all the discs to, and another post that can be utilized during the move will be given as the program input.
3. The objective is to produce the correct moves (the smallest number of movements) for the puzzle.

|  |  |
| --- | --- |
| Example input | Expected output |
| 2 2 3 1 | 2->1  2->3  1->3 |
| 3 3 1 2 | 3->1  3->2  1->2  3->1  2->3  2->1  3->1 |

Examples:

The first input: There are 2 discs. At the beginning, the discs are on Post 2. The objective is to move the discs to Post 3. Post 1 can be utilized during the movement.

The first output:

2->1: move the top disc from Post 2 to Post 1

2->3: move the top disc from Post 2 to Post 3

1->3: move the top disc from Post 1 to Post 3.

Similarly for the second example input and output.

**Hint**: in the test, you can assume the number of discs is no larger than 10.

## Program II:

Write a Java program (ClassStudents.java) according to the following requirement.

1. Create a class “Student” which is used to record a student’s information including his/her name, gender (male or female), birth year (1900 to 2022), and phone number. (Whether to use a “class” is optional)
2. Your program (ClassStudents.java) reads a few students’ information from the console input.
3. Perform information retrieval based on a given student’s name, and output his/her record.

|  |  |
| --- | --- |
| Example input | Expected output |
| 6  Mike, Male, 2003, 13910000000  Jack, Male, 2004, 13500000001  Rosemary, Female, 2004, 13661111111  Kate, Female, 2005, 18777519999  Mike, Male, 2005, 17999999999  Alexandre, Female, 2002, 19999999999  3  Mike  Jack  Alex | Mike: Male, 2003, 13910000000  Mike: Male, 2005, 17999999999  Jack: Male, 2004, 13500000001  Alex: None |

In the example, the first line of input is the number of students to be input, which is 6.

The next six lines are students’ information (name/gender/ year/phone), separated by a comma.

The next one line is the number of students to be searched (3 in this example), followed by the names to be searched.

For the search “Mike”: the information of two “Mike” in the input will be output to the console, in the order of the input.

For the search “Jack”: the information of one “Jack” in the input will be output to the console.

For the search “Alex”: output “None” since the name does not appear in the input.

Note: in the search results, “:” is used to separate the name and other information.

## Program Three:

Write a Java Program (TestMathExpr.java) with the following **Function A**, or with both **Function A** and **Function B**.

1. **Function A is the basic requirement.** It evaluates the value of math expressions, and outputs an integer value.
2. Each math expression includes (see the example below):
   1. numbers (integers and doubles);
   2. (no more than five) operators of “+” (addition), “-“ (subtraction), “\*” (multiplication) and “/” (division);
   3. possibly blank space.
3. All expressions are valid. The output is an integer value after rounding.

|  |  |
| --- | --- |
| Example input | Expected output |
| 1+2.0  -3+4/ 2.5+3.7  -3+4/2.5+3.9  1.2-3.5\*5.2-13.2  1.2-3.5\*5.2-13.7  2.3\*5\*7 - 12\*9/8  … | 3  2  3  -30  -31  67  … |

1. **Function B is a bonus requirement.** Each math expression includes (see the example below):
   1. numbers (integers and doubles);
   2. (no more than ten) operators of “+” (addition), “-“ (subtraction), “\*” (multiplication) and “/” (division);
   3. (no more than ten) functions including “sin” (sine function), “cos” (cosine function), “tan” (tangent function) and “sqrt” (square root function).
   4. “(“ and “)” (brackets);
   5. possibly blank space.
2. All the expressions are valid. So you don’t need to consider the invalid cases such as division by zero.

|  |  |
| --- | --- |
| Example input | Expected output |
| 1+2.0\*sin(37+(25\*3))  (2+ 3.50)\*4\*sqrt(sin(1.5))  -3+4/ (2.5+3.7)  (-3+4)/2.5+3.9  1.2-3.5\*5.2-13.2  1.2-3.5\*5.2-13.7  2.3\*5\*7 - 12\*9/8  -sin(3.5-sqrt(4)) + cos(tan(2.5))  … | -1  22  -2  4  -30  -31  67  0  … |

**Note:** Each submission is expected to strictly follow the following template to implement the required function by modifying the **parse()** function/method.

Text

Description automatically generated