COMP 202

Tutorial

Method recap

- Method header: e.g public static double abs(double aNumber)
- First *double* is the return type of the method, which can be void. If a return value is needed it must be declared here in the header. A *return var*; statement will end the method and return the result
- *Abs* is the name of the method, and you call the method using the name and parameters, e.g *double foo = abs(bar)*. You can think of method as a function f(x) where f is name, x is parameters, and you might expect this return you a result after certain calculation.
- *(double aNumber)* is the input parameter of the function. The method may take any number of parameters of any type, and the type of parameters should all be declared within this ( ), separated by ",". The variable name *aNumber* is the name of the input parameter as a variable within the scope of the method. This can be used directly as a variable in the method.

IF-ELSE recap

```
If( condition){
        //do something
}
else if (condition){
        //do something
}
else{
        //do something
}
```

If-else is used to control the flow of the execution of the code
The code with {} will only be executed if the condition in the () is met.
No “;” after (condition)
Else if is only checked when the previous conditions are not met
The condition is a Boolean (or any logic operation resulting a Bool
&&: AND only true&&true = true
||:OR true if any of the operator is true
!:NOT !true=false, !false=true

Casting recap

- Use (type) to cast a value/variable to another type
- (int) 3.96 = 3. Casting from double to integer
- (double) 3 = 3.0 Casting from integer to double
- For example, when you are dividing two integer and you want to have a floating number result instead of rounded result, you may want to use casting to cast one variable to double

e.g double quotient = (double)x/y

LOOP recap

- Loop: repetitively execute a block of code until a certain condition is met.
- While loop: while(condition){//do something)}
- In order to avoid infinite loop (if not intended), you should make change in the { } so that the condition will be met after a certain number of run.

E.g

```
Int I = 0;
while(i<10){
        System.out.println(i);
        i++;
}
```

This code will run 10 times before i reaches 10

Loop recap

```
For (int i=0;i<10;i++){
//do something
}
```

In the for loop, you have a () with 3 semicolons separated part

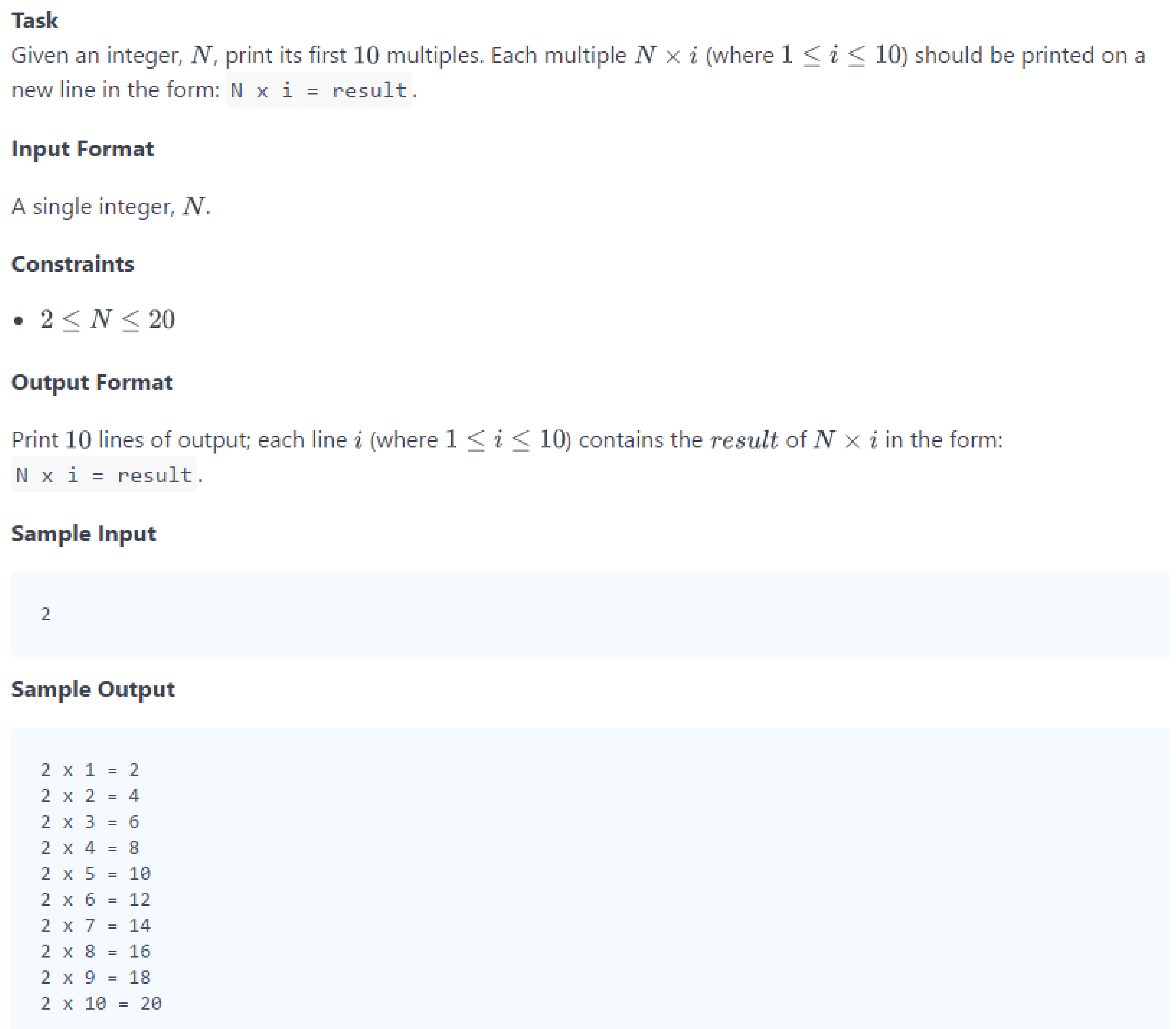
The first part int i=0 is used to declare a variable you want to use for condition, this can be initialized to anything, e.g int I = 10000

Second part is your condition to be checked

Third part is making change to your variable so condition will be eventually not met

Any of these part can be empty, this is equivalent to the while loop, only that you isolated the condition related statement in the ( )

Exercises



Task
Given an integer, N, print its first 10 multiples. Each multiple $N \times i$ (where $1 \leq i \leq 10$) should be printed on a new line in the form: `N x i = result`.

Input Format

A single integer, N.

Constraints

- $2 \leq N \leq 20$

Output Format

Print 10 lines of output; each line i (where $1 \leq i \leq 10$) contains the $result$ of $N \times i$ in the form:
`N x i = result`.

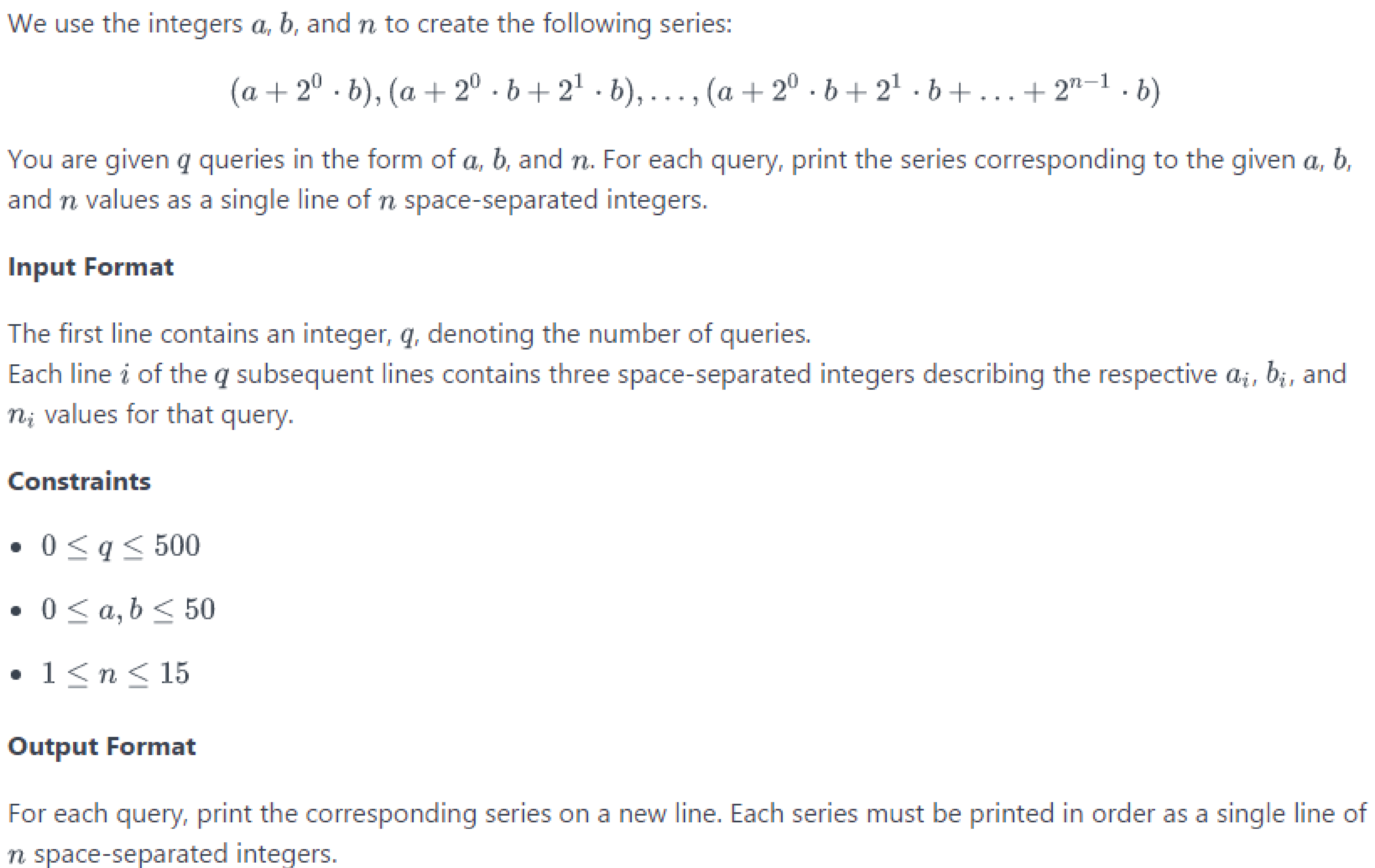
Sample Input

```
2
```

Sample Output

```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
```

2



We use the integers a, b, and n to create the following series:

$$(a + 2^0 \cdot b), (a + 2^0 \cdot b + 2^1 \cdot b), \ldots, (a + 2^0 \cdot b + 2^1 \cdot b + \ldots + 2^{n-1} \cdot b)$$

You are given q queries in the form of a, b, and n. For each query, print the series corresponding to the given a, b, and n values as a single line of n space-separated integers.

Input Format

The first line contains an integer, q, denoting the number of queries.
Each line i of the q subsequent lines contains three space-separated integers describing the respective a_i, b_i, and n_i values for that query.

Constraints

- $0 \leq q \leq 500$
- $0 \leq a, b \leq 50$
- $1 \leq n \leq 15$

Output Format

For each query, print the corresponding series on a new line. Each series must be printed in order as a single line of n space-separated integers.

Sample Input

```
2
0 2 10
5 3 5
```

Sample Output

```
2 6 14 30 62 126 254 510 1022 2046
8 14 26 50 98
```

Explanation

We have two queries:

1. We use $a = 0$, $b = 2$, and $n = 10$ to produce some series $s_0, s_1, \ldots, s_{n-1}$:

   - $s_0 = 0 + 1 \cdot 2 = 2$
   - $s_1 = 0 + 1 \cdot 2 + 2 \cdot 2 = 6$
   - $s_2 = 0 + 1 \cdot 2 + 2 \cdot 2 + 4 \cdot 2 = 14$

   ... and so on.

   Once we hit $n = 10$, we print the first ten terms as a single line of space-separated integers.

2. We use $a = 5$, $b = 3$, and $n = 5$ to produce some series $s_0, s_1, \ldots, s_{n-1}$:

   - $s_0 = 5 + 1 \cdot 3 = 8$
   - $s_1 = 5 + 1 \cdot 3 + 2 \cdot 3 = 14$
   - $s_2 = 5 + 1 \cdot 3 + 2 \cdot 3 + 4 \cdot 3 = 26$
   - $s_3 = 5 + 1 \cdot 3 + 2 \cdot 3 + 4 \cdot 3 + 8 \cdot 3 = 50$
   - $s_4 = 5 + 1 \cdot 3 + 2 \cdot 3 + 4 \cdot 3 + 8 \cdot 3 + 16 \cdot 3 = 98$

   We then print each element of our series as a single line of space-separated values.