CSDC101.ZC11 Fundamentals of Computing
Laboratory Exercise 3

PERFECT SCORE: 10/10

PROBLEMS

1) **Even** (3 points)

Lina has two numbers **a** and **b**. She wants you to print all even numbers from **a** and **b**, including **a** and **b**. That is, if they're even.

INPUT FORMAT
Input begins with an integer **N** indicating how many **a** and **b** pairs follow. Succeeding lines are pairs of integers **a** and **b**, separated by a space.

OUTPUT FORMAT
For each pair, display all even numbers from **a** and **b**, inclusive. Separate each with a single space. Printed even numbers for every pair must be in its own line.

CONSTRAINTS
```
1 <= N <= 100
1 <= a <= 50
51 <= b <= 100
```

SAMPLE INPUT
```
2
40 60
50 51
```

SAMPLE OUTPUT
```
40 42 44 46 48 50 52 54 56 58 60
50
```

2) **Programming is Fun v1** (3 points)

Saab is reminding herself that programming is fun. She keeps repeating "Programming is fun!" to herself so she won't forget. The number of times she'd repeat is based on an integer **n** she randomly picks.

INPUT FORMAT
Input contains an integer **n** indicating how many times "**Programming is fun!**" should be printed.

OUTPUT FORMAT
Print "**Programming is fun!**" without the quotation marks **n** times. Separate each with a newline. If the value of **n** is invalid (refer to the constraints for **n**), the program prints nothing.

CONSTRAINTS

```
1 <= n <= 100
```

SAMPLE INPUT

```
3
```

SAMPLE OUTPUT

```
Programming is fun!
Programming is fun!
Programming is fun!
```

3) **Programming is Fun v2 (3 points)**

** A variation of Programming is Fun v1. Check the input and output format below. **

Saab is reminding herself that programming is fun. She keeps repeating "Programming is fun!" to herself so she won't forget. The number of times she'd repeat is based on an integer **n** she randomly picks.

INPUT FORMAT

Input contains and integer **n** indicating how many times "Programming is fun!" should be printed.

OUTPUT FORMAT

Print "**Programming is fun!**" without the quotation marks **n** times. Separate each with a newline. If the value of **n** is invalid (refer to the constraints for **n**), print "**Programming is fun!**" **once**.

CONSTRAINTS

```
1 <= n <= 100
```

SAMPLE INPUT

```
1
```

SAMPLE OUTPUT

```
Programming is fun!
```

4) **se7en (3 points)**

Tubby likes writing the number 7 as "se7en", including all its multiples. In a list of numbers given by Tubby, find all the 7s and all multiples of 7 and replace them with "se7en".

INPUT FORMAT

Input contains an arbitrary list of integers, each in a new line. The values of the integers are from -1000 to 1000, inclusive. The input is terminated by any number outside of the accepted values.

OUTPUT FORMAT

Display the integers to the screen with all the "7" and all multiples of 7 replaced with "se7en". Integers should be separated by a single space.

SAMPLE INPUT

```
1
7
49
33
16
700
1001
```

SAMPLE OUTPUT

```
1 se7en se7en 33 16 se7en
```

5) **Bank (5 points)**

Lulu is monitoring her bank transactions in a particular account. She asked you to write her a program that will automate the computations for her. She wants a simple program that would keep track of the account's maintaining and current balances. It should also be able to update the current balance based on the following transactions:

a) Deposit – add amount to current balance.
b) Withdraw – deduct amount from current balance.

She likewise tells you that withdrawals should only be allowed if it won't cause the current balance's amount go below the maintaining balance.

INPUT FORMAT

Input begins with an integer **`M`** denoting the maintaining balance for the account
The second line of input is an integer **`B`** denoting her account's current balance.
The third line contains an integer **`N`** indicating how many transactions follow.
Succeeding lines are **`N`** character **`c`** and integer **`a`** pairs, one in each line, denoting the transactions that will be done to Lulu's bank account, where **`c`** is either 'D' or 'W' for "Deposit" and "Withdrawal", respectively; and **`n`** is the amount to be deposited or withdrawn.

OUTPUT FORMAT

For each transaction, display the updated current balance. If withdrawing an amount will cause the current balance's value to go below the maintaining balance, print "**Cannot withdraw!**". Output should be separated by a new line.

CONSTRAINTS

```
500 <= M <= 10000
500 <= B <= 100000
1 <= N <= 100
C is either 'D' or 'W'
1 <= a <= 10000
```

SAMPLE INPUT

```
500
500
3
D 50
W 600
```

```
D 100
```

SAMPLE OUTPUT

```
550
Cannot withdraw!
650
```