ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)

ORGANISATION OF ISLAMIC COOPERATION (OIC)

Department of Computer Science and Engineering (CSE)

CSE 4108: Structured Programming I Lab
Lab 5, Section 1A

Objectives

- Exploring Conditional Statements
- Introduction to Loops

Guidelines

- All source codes (.c files) in the naming format `ID_Lab4_TaskN.c`, eg. `230041101_Lab5_Task4.c`.
- Ensure that your program produces the correct output for all sample cases.
- Source codes must be properly indented
- Screenshots are not required.

.

Task 1

Write a program that takes two integer inputs from the user: a starting number and an ending number. The program should then find and calculate the sum of all numbers between (and including) the starting and ending numbers that are both **odd** and **divisible by 5.**

Input

The input consists of two integers, a and b, $(1 \leq a \leq b \leq 1000)$
Note: For this task, and following tasks, you do not need to manually check whether the input follows these constraints or not - it is guaranteed that the provided input **will** follow these constraints.

Sample Executions

Sample Execution 1

```
Enter two numbers:  4 16
Sum is:  20
```

Sample Execution 2

```
Enter two numbers:  1 1000
Sum is:  50000
```

Task 2

Write a C program that continuously number inputs from the user, one at a time. The program should keep taking inputs until the user enters a value that is either 0 or negative. When a 0 or negative number is entered, the program should stop accepting inputs and do the following:

- If any positive numbers were entered, the program should display the smallest positive number among all the inputs.
- If no positive numbers were entered (i.e., the first input itself is 0 or a negative number), the program should display `NONE`.

Sample Executions

Sample Execution 1

```
Enter a number:  13.54
Enter a number:  2
Enter a number:  1.75
Enter a number:  3.14
Enter a number:  0
Smallest number is:  1.75
```

Sample Execution 2

```
Enter a number:  -8
Smallest number is:  NONE
```

Task 3

Write a program that takes a single integer input from the user and checks whether the number is a prime number or not.
A prime number is a whole number greater than 1 that has no divisors other than 1 and itself.

Input

The input consists of a single integer n, such that $2 \leq n \leq 10^{12}$.

Note: the `int` datatype can only read numbers up to roughly 2×10^9, use `long long` instead.

Output

If the number is prime, output `YES`, otherwise output `NO`.

Sample Test Cases

Input

```
13
```

Output

```
YES
```

Input

```
25
```

Output

```
NO
```

Input

```
35
```

Output

```
NO
```

Input

```
999999999
```

Output

```
NO
```

Input

```
9555312829
```

Output

```
YES
```

Task 4

Write a program that takes an integer n as input, and create a K-shape consisting of n lines using **your name** in uppercase. The output must consist of n lines.

Use **width specifiers** for left and right justification to align the names properly. The spacing between the names must exactly match the provided output format given.
You are not allowed to use any whitespaces in your printf() function in this task.

Input

The input consists of a single integer n, such that $7 \leq n \leq 79$, and n is odd.

Output

The shape `K` written using **your name**. Check the sample outputs for clarity.

Sample Test Cases (Using the name Ditto)

Input

```
7
```

Output

```
DITTO     DITTO
DITTO   DITTO
DITTO DITTO
DITTO
DITTO DITTO
DITTO   DITTO
DITTO     DITTO
```

Input

```
15
```

Output

```
DITTO             DITTO
DITTO           DITTO
DITTO         DITTO
DITTO       DITTO
DITTO     DITTO
DITTO   DITTO
DITTO DITTO
DITTO
DITTO DITTO
DITTO    DITTO
DITTO      DITTO
DITTO        DITTO
DITTO          DITTO
DITTO            DITTO
DITTO              DITTO
```