CHIANG MAI UNIVERSITY
College of Arts, Media and Technology
1st Semester / Academic Year 2025
Fundamental of Programming Logic in Digital Industry

Lab Assignment 06 : While-Statement and Program Tracing

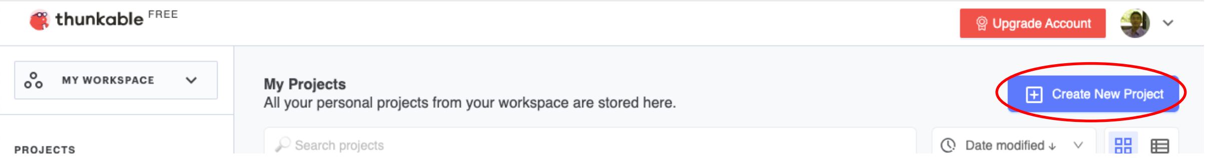
Name ....กันต์ธีร์ วารีสอาด.......................... Student ID ....Section.........1......

Objectives:

1) Students understand the logic of repetition programming.
2) Students can program using while-statement in Thunkable
3) Students can use the trace table to do debugging the program.

Get Start

1. Go to https://thunkable.com/ and login.
2. From “**My Project**” page, Click on “**Create New Project**” button. Once the window pop up, input new project name as “**Lab06**” and select category of project as “**Education**”.

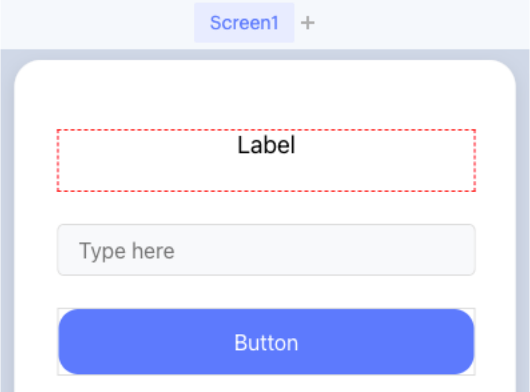


All your personal projects from your workspace are stored here.

Repetition Structure (while-loop) in Thunkable

Example 1

1. In the “**Design**” view, create a label, a text input and a button on the app interface.

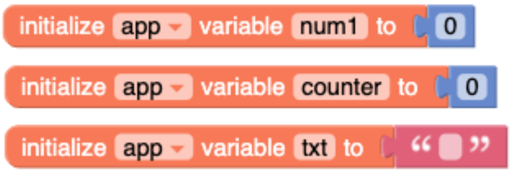


2. To begin, three global variables, named “**num1**”, “**counter**”, and “**txt**”, must be declared and initialized with the value 0, 0, and "", respectively. Thus,
   2.1 Drag three “**initialize**” bocks from Variable tab into the block design console.

2.2 Drag two “**number**” blocks from Math tab into the block design console.

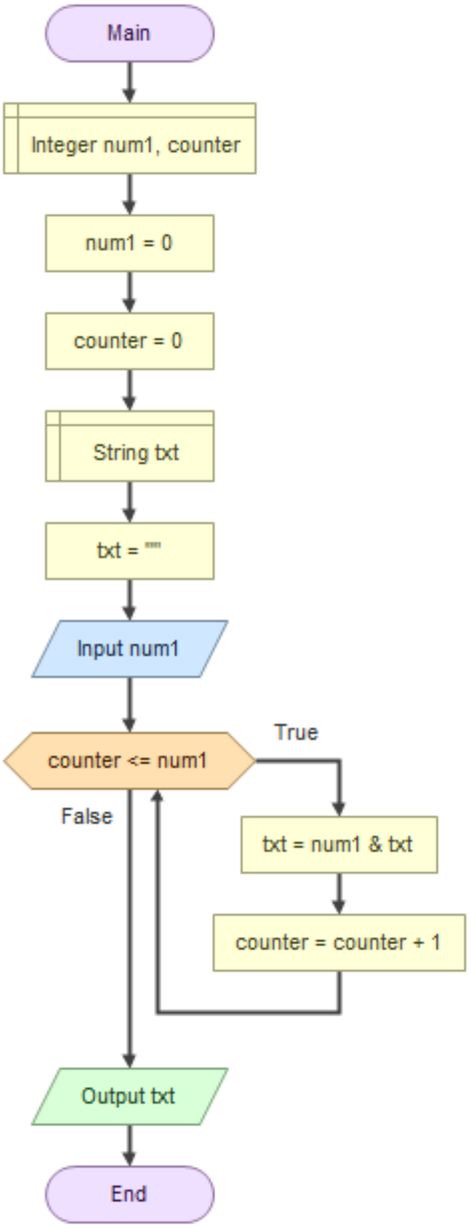
2.3 Drag a “**word**” blocks from Text tab into the block design console.

2.4 Form the blocks as following:

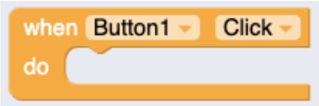


```
initialize app variable num1 to 0
initialize app variable counter to 0
initialize app variable txt to " "
```

3. In the “**Click event**” of button1, configure the loop component as follows:
   1) **Stop condition** – repeat the body as long as the counter is lesser or equal to “**num1**”.
   2) **Loop body** – Assign the value of variable “**num1**” to the “**txt**” variable.
   3) **Update statement** – add 1 to the counter.

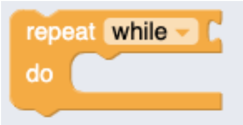


4. In the “**Blocks**” view,
   4.1 Click on **Button1** UI component and drag the “**Click event**” block into the block design console.



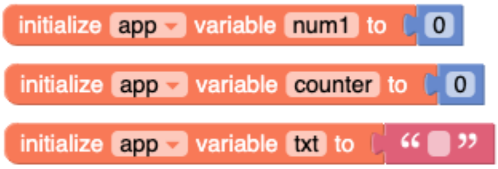
```
when Button1 Click
do
```

   4.2 Click on **Control** tab and drag the following blocks into the block design console.

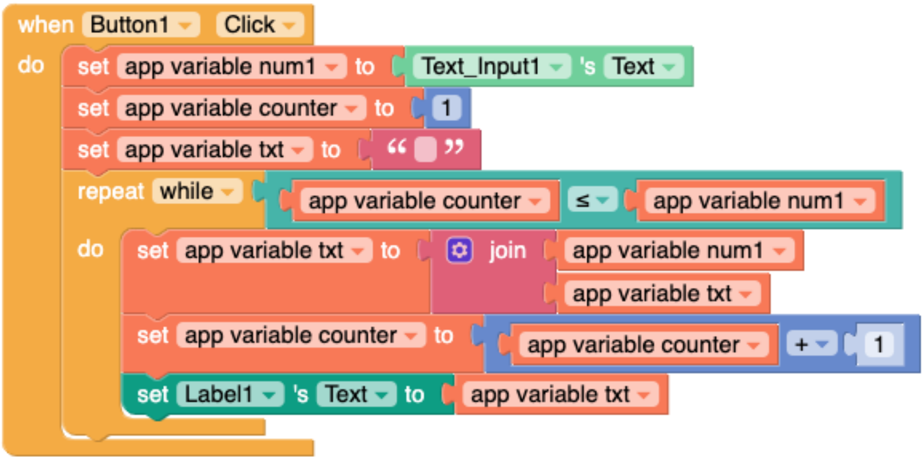


```
repeat while
do
```

   4.3 Add necessary blocks into the block design console and form the set of blocks according to the flowchart provided above as the following:



```
initialize app variable num1 to 0
initialize app variable counter to 0
initialize app variable txt to " "
```



```
when Button1 Click
do  set app variable num1 to Text_Input1 's Text
    set app variable counter to 1
    set app variable txt to " "
    repeat while  app variable counter ≤ app variable num1
    do  set app variable txt to join  app variable num1
                                      app variable txt
        set app variable counter to  app variable counter + 1
        set Label1 's Text to app variable txt
```

5. From the program which implementing the logic of the **while** statement above, complete the following table:

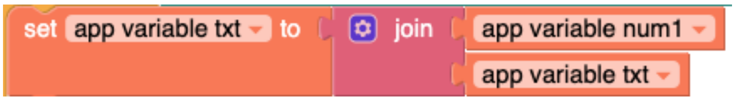
Input in Text_Input1	**Text in Label1**
5	55555
7	7777777
10	10101010101010101010

Hint: Run the program to test the answer.

Example 2

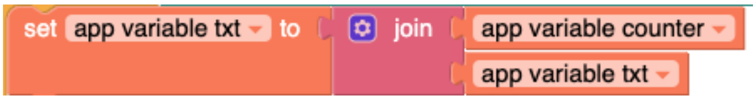
This example will extend the **Example 1** to demonstrate the operation of the loop.

1. Use the same interface as **Example 1**.
2. In the “**Blocks**” view, change the loop body from adding the variable “**num1**” to the variable “**txt**”.



```
set app variable txt to join app variable num1
                             app variable txt
```

to adding the variable “**counter**” to the variable “**txt**”.



```
set app variable txt to join app variable counter
                             app variable txt
```

3. Preview the program and complete the following table:

Input in Text_Input1	Text in Label1
5	54321
7	7654321
10	10987654321

Problem Set

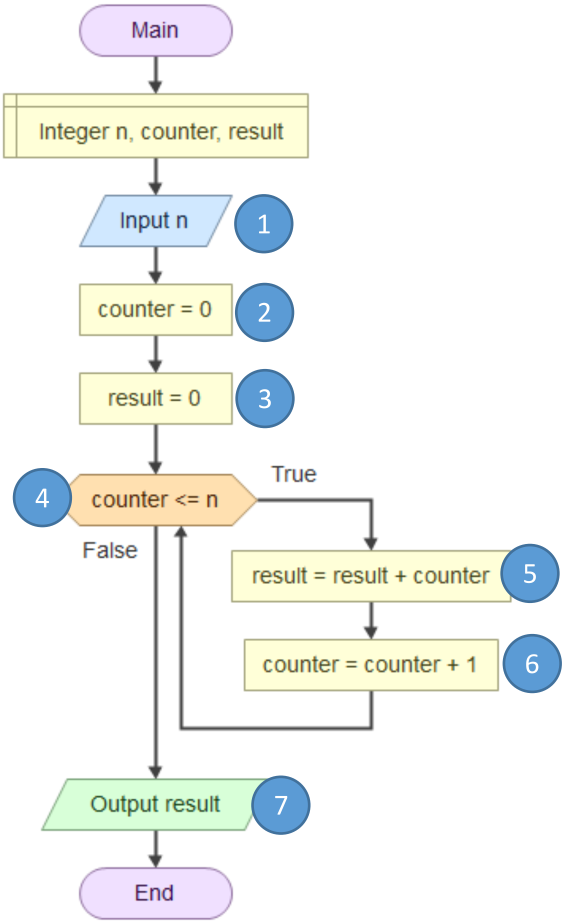
1. Given the following pseudocode,

```
GET num1,num2
counter = 0
result = 1
WHILE counter < num2
    result = result * num1
    counter = counter + 1
ENDWHILE
DISPLAY result
```

If the user input 2 for “**num1**” and 3 for “**num2**”, complete the following the tracing table.

No.	Variable/Logic	Statement/Operation	Value
1	num1	GET num1	2
2	num2	GET num2	3
3	counter	counter = 0	0
4	result	result = 1	1
5	counter < num2	0 < 3	TRUE
6	result	result = 1 * 2	2
7	counter	counter = 0 + 1	1
8	counter < num2	1 < 3	TRUE
9	result	result = 2 * 2	4
10	counter	counter = 1 + 1	2
11	counter < num2	2 < 3	TRUE
12	result	result = 4 * 2	8
13	counter	counter = 2 + 1	3
14	counter < num2	3 < 3	FALSE
15	result	DISPLAY result	8

2. Given the following flowchart,

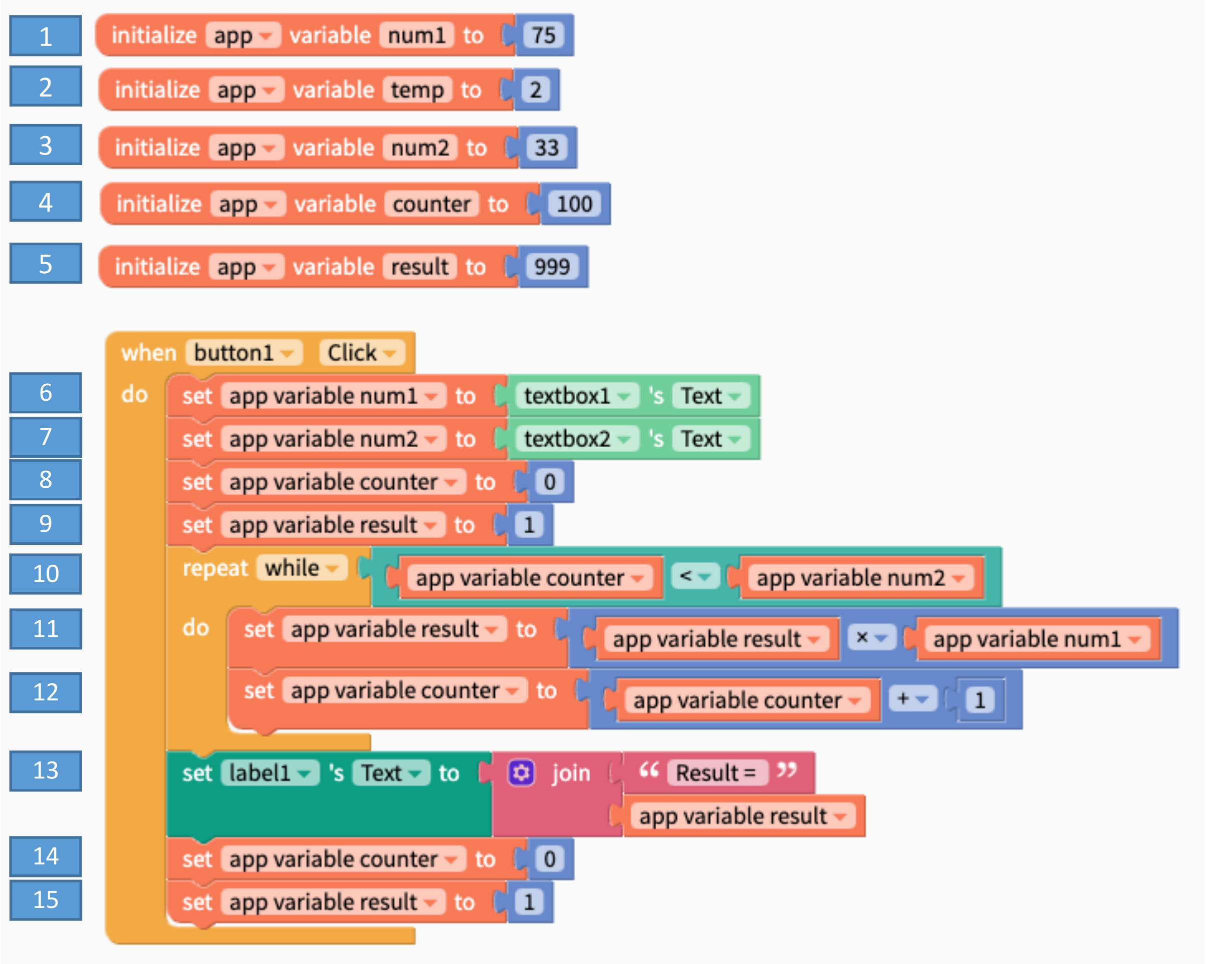


Complete the following tracing table if the user input **3** for variable **n**.

No.	**Variable/Logic**	**Statement/Operation**	**Value**
1	n	GET n	3
2	counter	counter = 0	0
3	result	result = 0	0
4	counter <= n	0 <= 3	TRUE
5	result	result = 0 + 0	0
6	counter	counter = 0 + 1	1
7	counter <= n	1 <= 3	TRUE
8	result	result = 0 + 1	1
9	counter	counter = 1 + 1	2

No.	Variable/Logic	Statement/Operation	Value
10	counter <= n	2 <= 3	TRUE
11	result	result = 1 + 2	3
12	counter	counter = 2 + 1	3
13	counter <= n	3 <= 3	TRUE
14	result	result = 3 + 3	6
15	counter	counter = 3 + 1	4
16	counter	4 <= 3	FALSE
17	result	OUTPUT result	6

3. Given the following **Thunkable** set of blocks,



```
1   initialize app variable num1 to 75
2   initialize app variable temp to 2
3   initialize app variable num2 to 33
4   initialize app variable counter to 100
5   initialize app variable result to 999

    when button1 Click
    do
6       set app variable num1 to textbox1 's Text
7       set app variable num2 to textbox2 's Text
8       set app variable counter to 0
9       set app variable result to 1
10      repeat while app variable counter < app variable num2
        do
11          set app variable result to app variable result × app variable num1
12          set app variable counter to app variable counter + 1
13      set label1 's Text to join " Result = "
                                  app variable result
14      set app variable counter to 0
15      set app variable result to 1
```

If the user input **4** for "**num1**" and **4** for "**num2**", complete the following program tracing table.

No.	Variable/Logic	Statement/Operation	Value
1	counter	counter = 0	0
2	result	result = 1	1
3	counter <= num2	0 <= 4	TRUE
4	result	result = 1 * 4	4
5	counter	counter = 0 + 1	1
6	counter <= num2	1 <= 4	TRUE
7	result	result = 4 * 4	16
8	counter	counter = 1 + 1	2
9	counter <= num2	2 <= 4	TRUE

No.	Variable/Logic	Statement/Operation	Value
10	result	result = 16 * 4	64
11	counter	counter = 2 + 1	3
12	counter <= num2	3 <= 4	TRUE
13	result	result = 64 * 4	256
14	counter	counter = 3 + 1	4
15	counter <= num2	4 <= 4	TRUE
16	result	result = 256 * 4	1024
17	counter	counter = 4 + 1	5
18	counter <= num2	5 <= 4	FALSE
19	label1.Text	show result	1024

ALL PROBLEMS MUST USE WHILE-LOOP !!!!!!!

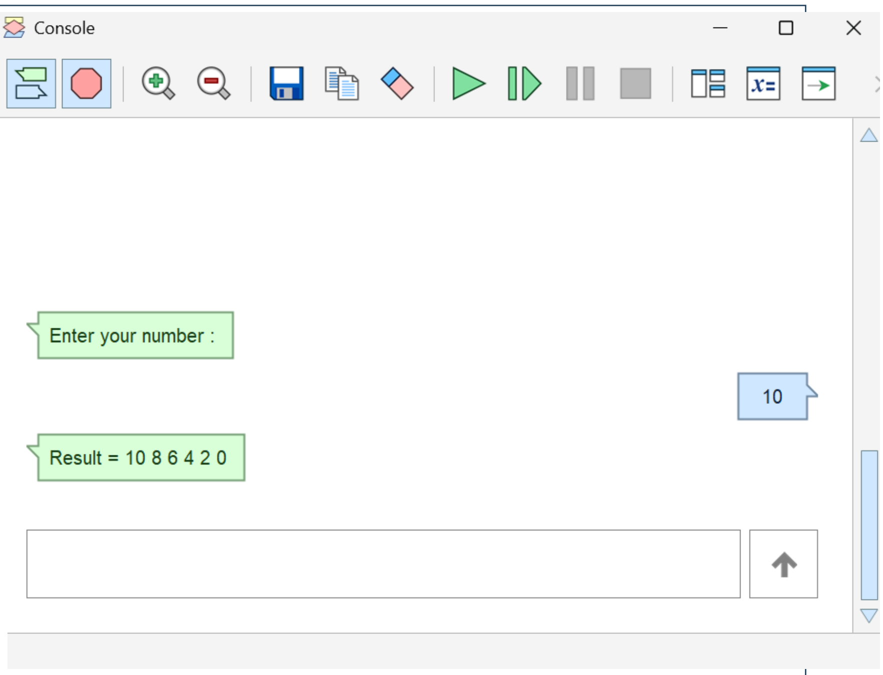
4. Develop a program with a textbox, a label, and a button on its interface to accept an integer from the user and output a series of numbers decreasing by 2 until the value reaches 0. The result's text will be display in the label after click on button.

   For example,

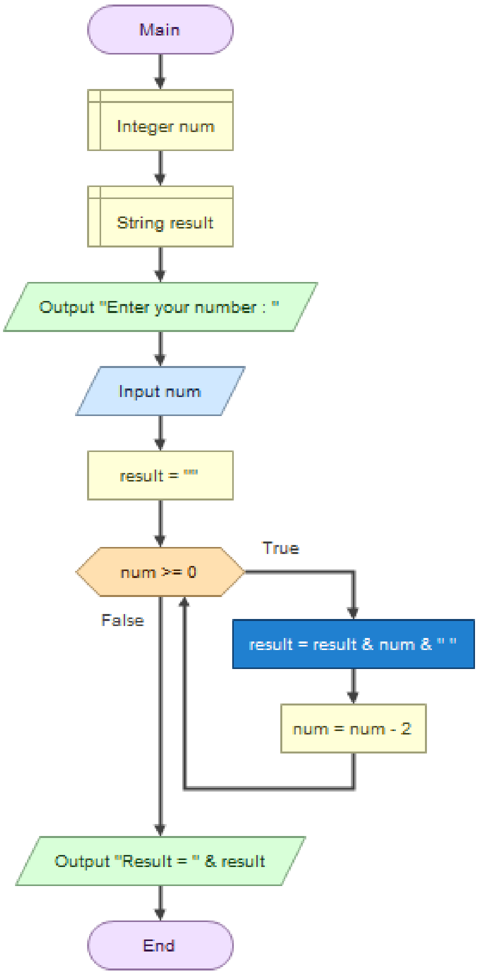
   If the user inputs **10**, the program will display **10 8 6 4 2 0**

   If the user inputs **13**, the program will display **13 11 9 7 5 3 1**

   4.1 Draw a flowchart of the program using **Flowgorithm**.

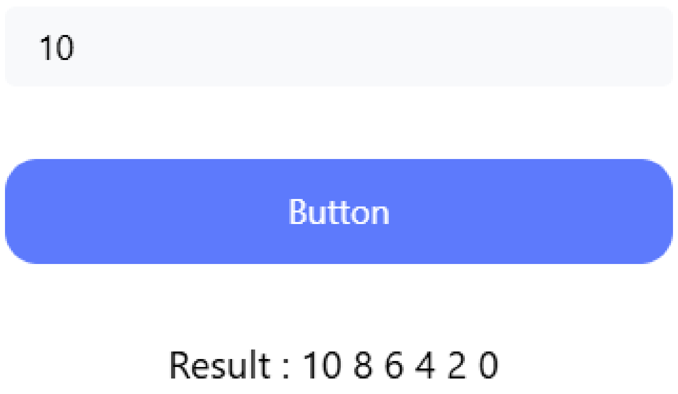


```
Enter your number :
                                                    10
Result = 10 8 6 4 2 0
```


Main
Integer num
String result
Output "Enter your number : "
Input num
result = ""
num >= 0
True
False
result = result & num & " "
num = num - 2
Output "Result = " & result
End

capture of Flowgorithm HERE!!

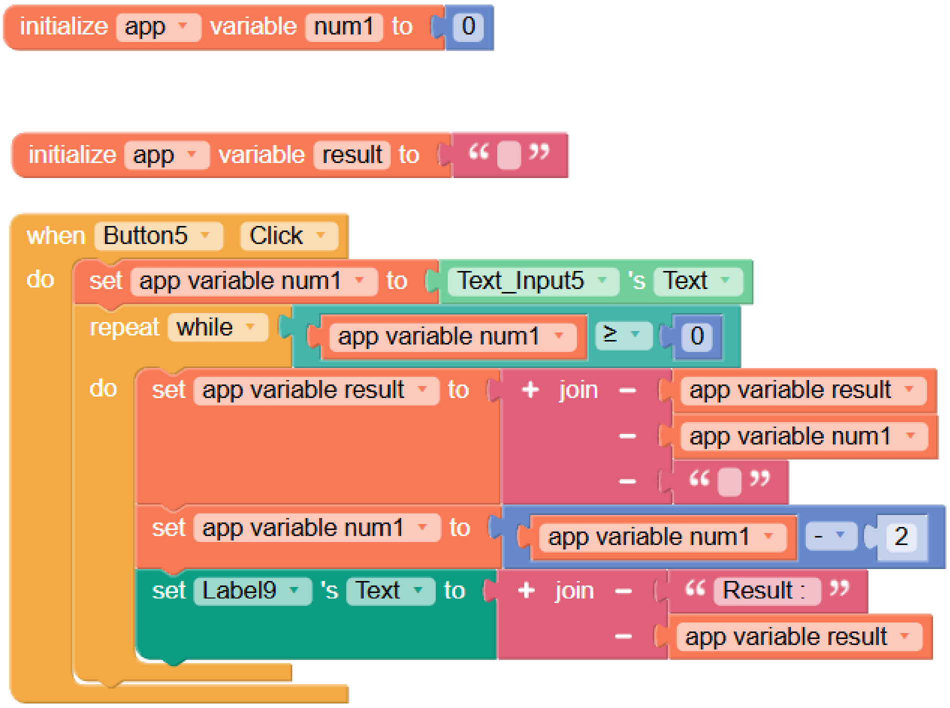
4.2 Develop the program in **Thunkable** (Always create new screen!!).



10

Result : 10 8 6 4 2 0

Insert screen capture of Thunkable HERE!!



```
initialize app variable num1 to 0

initialize app variable result to " "

when Button5 Click
do  set app variable num1 to Text_Input5 's Text
    repeat while  app variable num1 ≥ 0
    do  set app variable result to  join  app variable result
                                          app variable num1
                                          " "
        set app variable num1 to  app variable num1 - 2
    set Label9 's Text to  join  " Result : "
                                 app variable result
```

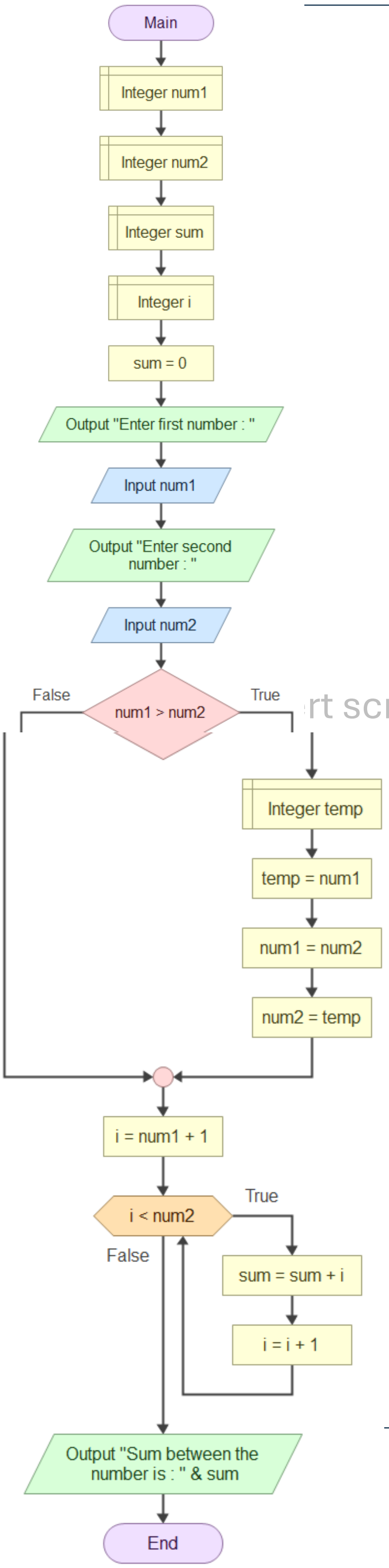
5. Develop a program that accepts two user input values and outputs the sum of the digits between them only (without the input numbers themselves). The result's text will be display in the label after click on button.

   For example,

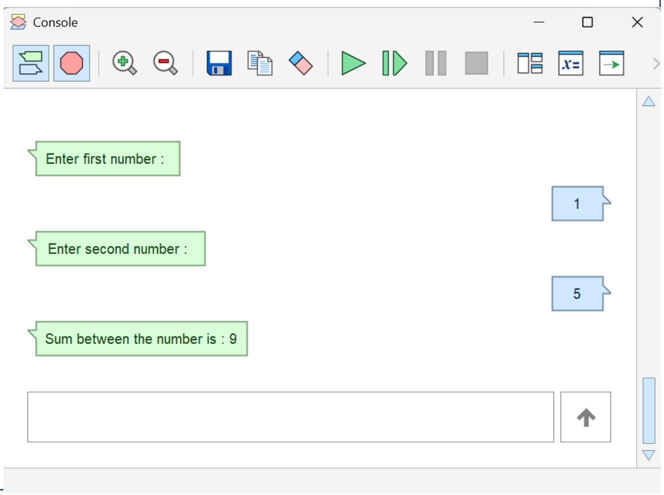
   If the user input **1** and **5**, the program will display **9** (2+3+4 = 9)

   If the user input **2** and **7**, the program will display **18** (3+4+5+6 = 18)

   5.1 Draw a flowchart or pseudocode of the program using **Flowgorithm**.

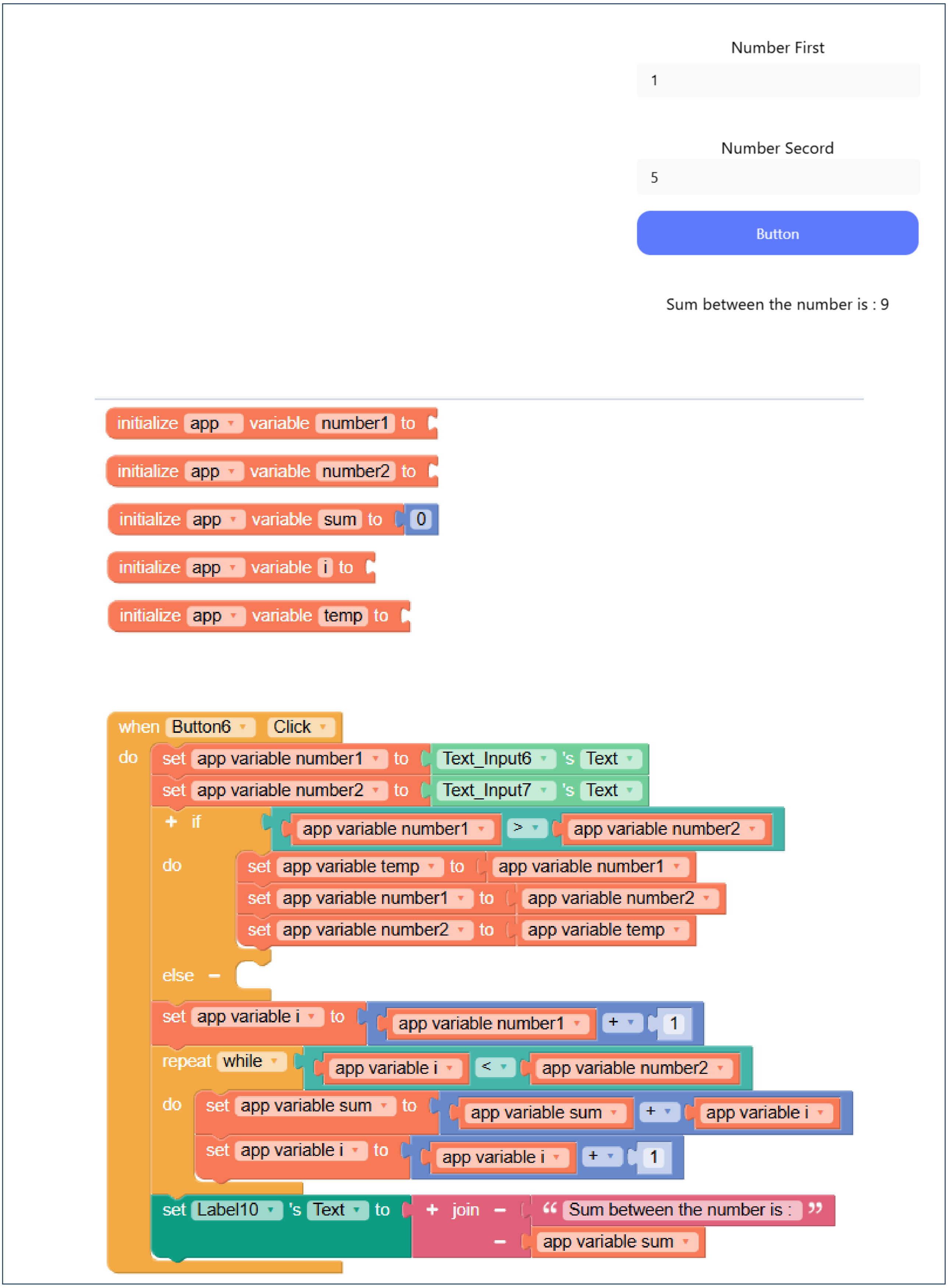


rt screen capture of Flowgorithm HERE!!



```
Enter first number :
                                        1
Enter second number :
                                        5
Sum between the number is : 9
```

5.2 Develop the program in **Thunkable** (Always create new screen!!).



Number First

1

Number Secord

5

Button

Sum between the number is : 9

```
initialize app variable number1 to
initialize app variable number2 to
initialize app variable sum to 0
initialize app variable i to
initialize app variable temp to

when Button6 Click
do  set app variable number1 to Text_Input6 's Text
    set app variable number2 to Text_Input7 's Text
    + if  app variable number1 > app variable number2
      do  set app variable temp to app variable number1
          set app variable number1 to app variable number2
          set app variable number2 to app variable temp
      else −
    set app variable i to app variable number1 + 1
    repeat while app variable i < app variable number2
    do  set app variable sum to app variable sum + app variable i
        set app variable i to app variable i + 1
    set Label10 's Text to + join − " Sum between the number is : "
                                  − app variable sum
```