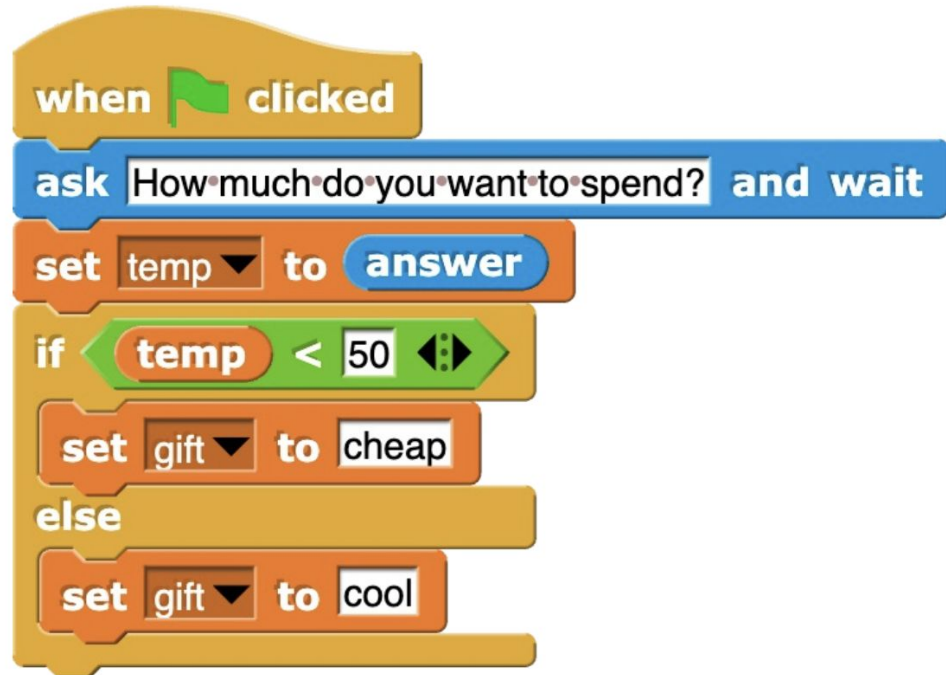


**Q: What is the value of gift after the block is run, assume user input is 100?**



iClicker

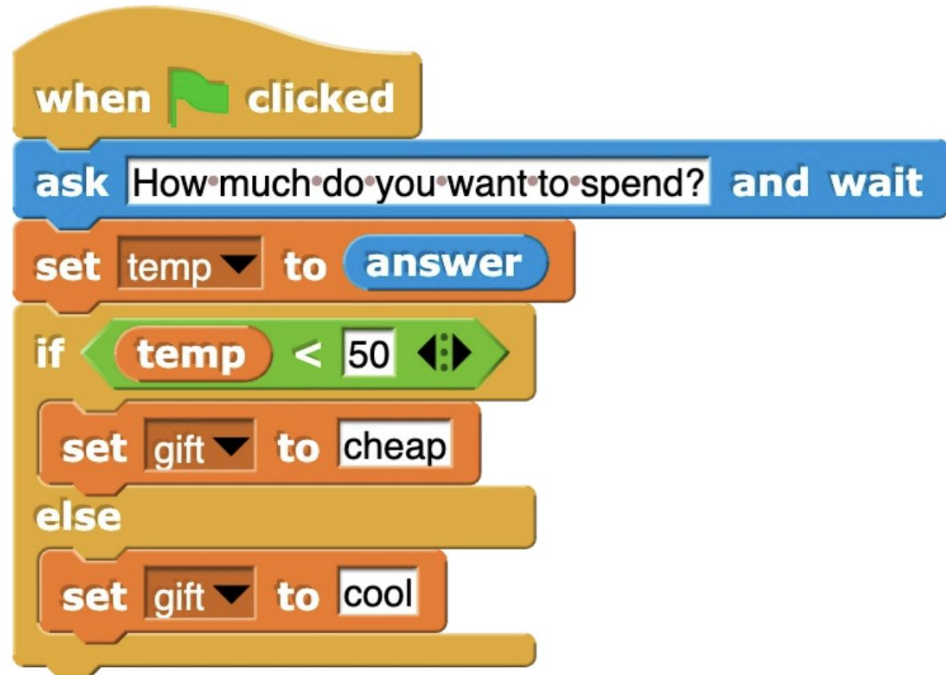
- A. 50
- B. 100
- C. 150
- D. cheap
- E. cool



**Q: What is the value of gift after the block is run, assume user input is 50?**



- A. 50
- B. 100
- C. 150
- D. cheap
- E. cool





# CPSC 100

## Computational Thinking

### Boolean Functions + Loops

Instructor: Parsa Rajabi  
Department of Computer Science  
University of British Columbia

# Agenda

- Learning Goals
- Course Admin
- Intro to Programming [Continued]
  - Boolean Functions/Expressions
  - Repeat Blocks → For Loops



# Learning Goals

After this week's lecture, you should be able to:

- Define **boolean expressions** and their role in Snap! Programming
- **Apply** boolean operations to control flow in programming
- Describe the concept of **loops** and **iteration** in programming
  - Differentiate between **finite loops** (repeat N times) and **condition-based loops** (repeat until)
- Use logical reasoning to **determine the output of given code**
  - Apply CT to **trace and evaluate** code snippets

# Course Admin

# Course Admin

- **Lab #4**

- The Evolution of Trust; <https://ncase.me/trust/>
- Review/Play Game **before your lab**
- Due on Thursday, Feb 6 at 11:59pm

- **Post-Class (PC) Quiz #3**

- Only 1 attempt, 60 minutes
- To be posted tonight; Due on Sunday, Feb 9 at 11:59pm

- **Midterm**

- All content covered from Week 1 Until end of Week 5 (this Friday)
- We will likely be writing in a **different room**; I will confirm soon
- Past Exams/Practice Questions to be released soon

# Boolean (or Logical) Function







# Boolean (or Logical) Function

At the very lowest level, computer circuitry is made of wires, and each wire is either **on or off**.

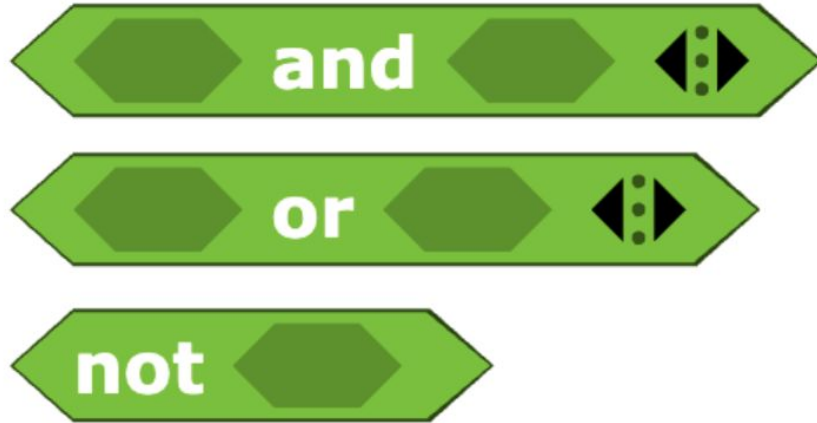
The only operations that can be performed at the lowest level are those that operate on **single-bit values**

0 or 1

on or off



# Boolean (or Logical) Function



Notice that both the blocks themselves and the input slots in the blocks are hexagonal.

**Boolean functions** take Boolean values (True or False) as **inputs** and report a new Boolean value as **output**.



# Boolean → Data Representation

**OR Blocks** | Evaluates to

 False

 True

 True

 True

**OR** evaluates to **true**, as long as **one operand evaluates to true**



# Boolean → Data Representation

**OR Blocks** | Evaluates to

 False

 True

 True

 True

**AND Blocks** | Evaluates to

 False

 False

 False

 True

**OR** evaluates to **true**, as long as one operand evaluates to true  
**AND** evaluates to **true**, only if all operands are true



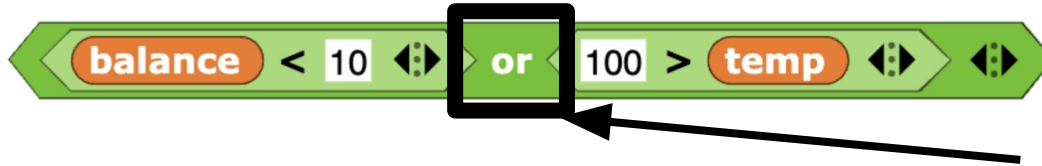
# Boolean Function Examples



balance and temp are variables



if balance is less than 10 **AND**  
if temp is greater than 100



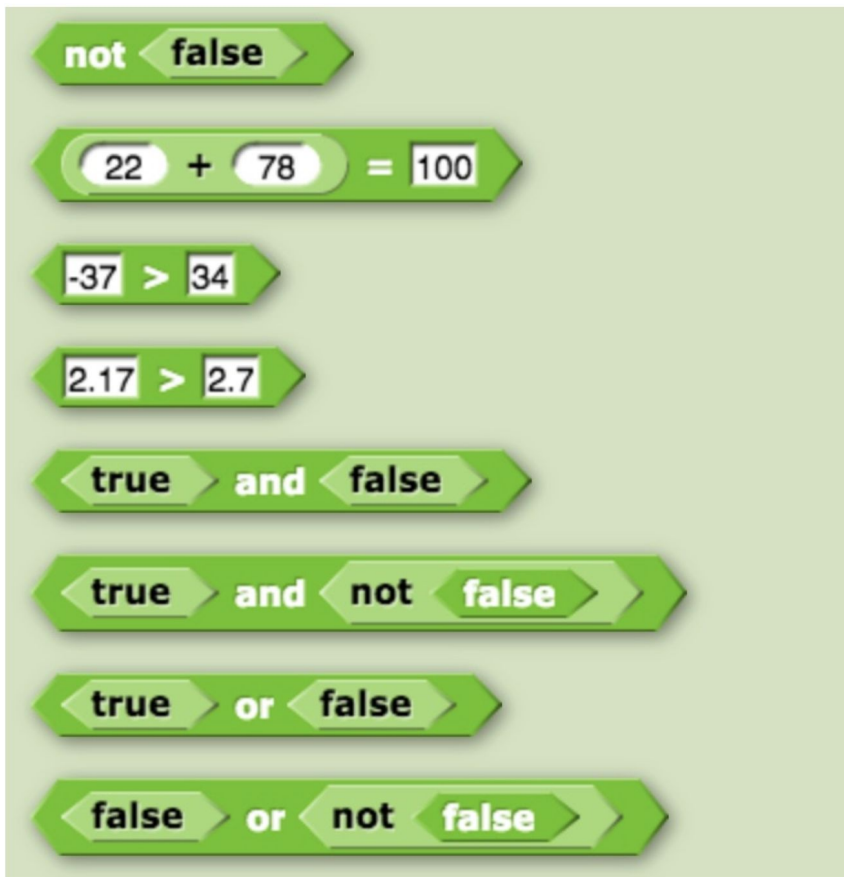
if balance is less than 10 **OR**  
if temp is greater than 100



if balance is **NOT** less than 10

# Mini-Activity

# What does each block evaluate to?



A collection of Scratch-style logic blocks arranged vertically on a light green background. The blocks are as follows:

- A green flag block with the text "not false".
- A green math block with the expression  $22 + 78 = 100$ .
- A green comparison block with the expression  $-37 > 34$ .
- A green comparison block with the expression  $2.17 > 2.7$ .
- A green logic block with the expression "true and false".
- A green logic block with the expression "true and not false".
- A green logic block with the expression "true or false".
- A green logic block with the expression "false or not false".

True?

OR

False?





# What does each block evaluate to?

True.

not false

True.

22 + 78 = 100

False.

-37 > 34

False.

2.17 > 2.7

False.

true and false

True.

true and not false

True.

true or false

True.

false or not false

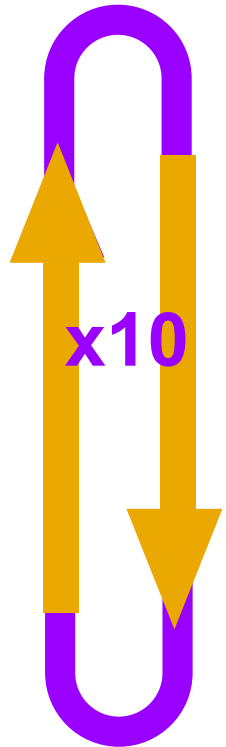
True?

OR

False?

# Iteration

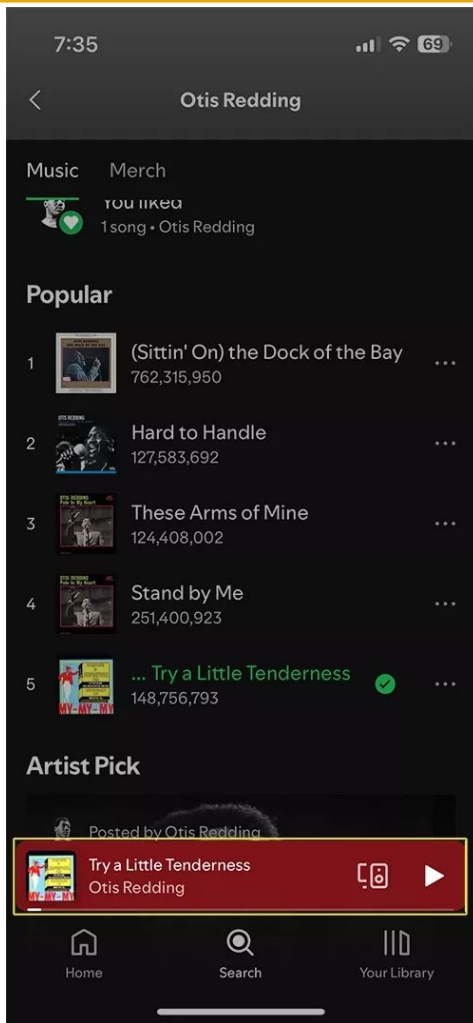
Allows the algorithm to repeat instructions.



**Repeat 10 times:**

1. Preheat oven (400° C)
2. Combine ingredients in bowl to form dough
3. Put dough into bread pan
4. If ingredients contain yeast, allow to sit at room temperature for 1 hour
5. Put bread pans into preheated oven and bake for 30 minutes





# Iteration

What if you want to do a task over and over again?

A **loop** allows you to do the same task over & over again, sometimes with a **stopping** condition, sometimes **forever**!





# Repeat Blocks

**Repeat** some code a finite number of time



**Repeat UNTIL** a particular condition has been met.

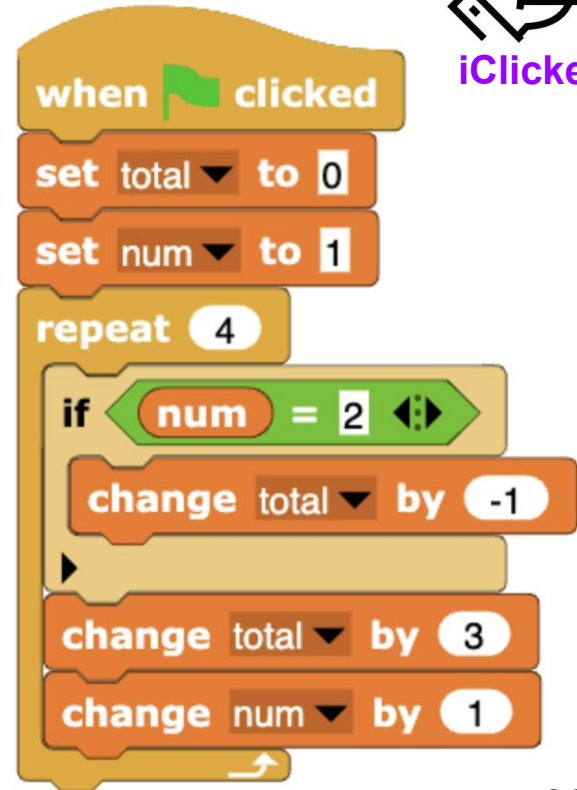
**If** the condition is never met, then, it goes on *forever*.



**Q: What is the value in **total** and **num** when this code block is run?**



- A. total = 11; num = 5
- B. total = 12; num = 5
- C. total = 9; num = 4
- D. total = 10; num = 5
- E. total = 11; num = 6



**Q: What is the value in **total** and **num** when this code block is run?**



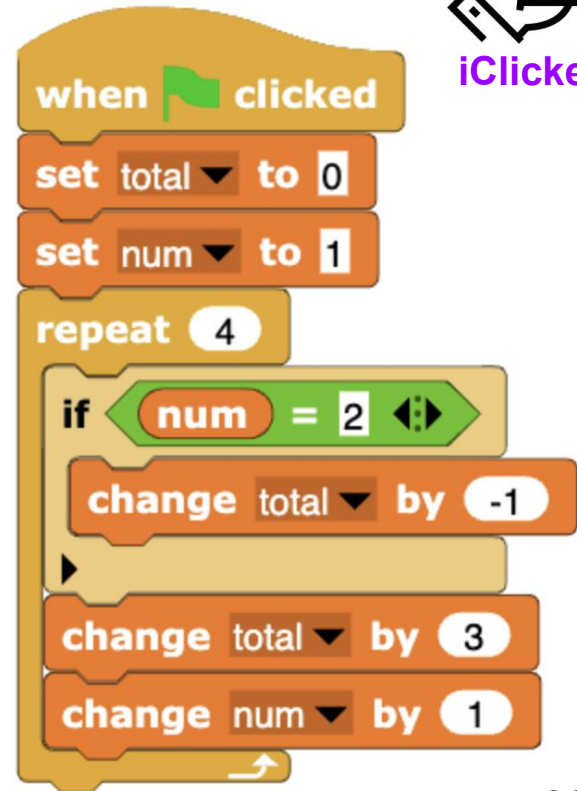
A. total = 11; num = 5

B. total = 12; num = 5

C. total = 9; num = 4

D. total = 10; num = 5

E. total = 11; num = 6



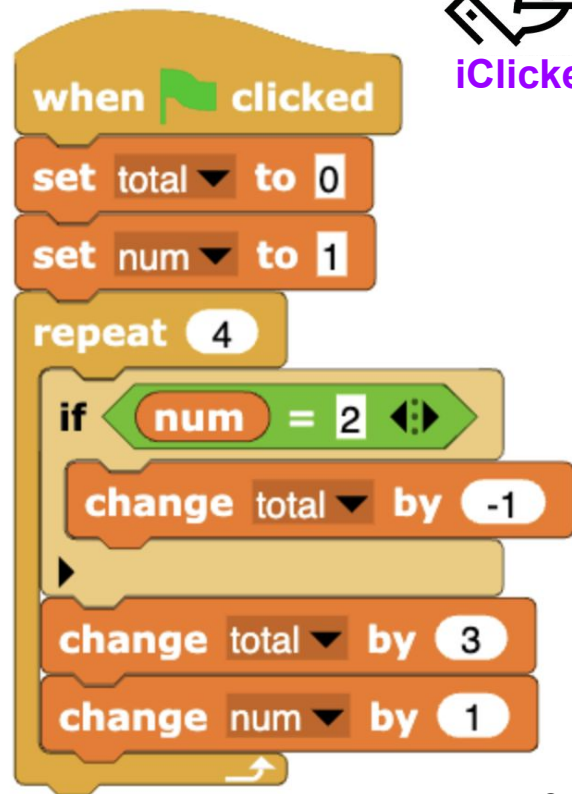


# Q: What is the value in **total** and **num** when this code block is run?



iClicker

Iteration	num (Before)	Condition Check (num = 2?)	Change to total (-1 if true)	total +3	total (After)	num +1	num (After)
1st	1	No (skip -1)	0	+3	3	+1	2
2nd	2	Yes (-1 applied)	$3 - 1 = 2$	+3	5	+1	3
3rd	3	No (skip -1)	5	+3	8	+1	4
4th	4	No (skip -1)	8	+3	11	+1	5



**Q: What is the value in *i* when the code is run, assuming user input = 3?**



- A. 1
- B. 2
- C. 3
- D. 4
- E. 5

```

when clicked
  ask Pick a number between 1 and 10 and wait
  set i to 1
  set total to 0
  repeat answer
    change total by i
    change i by 1
  
```

**Q: What is the value in *i* when the code is run, assuming user input = 3?**



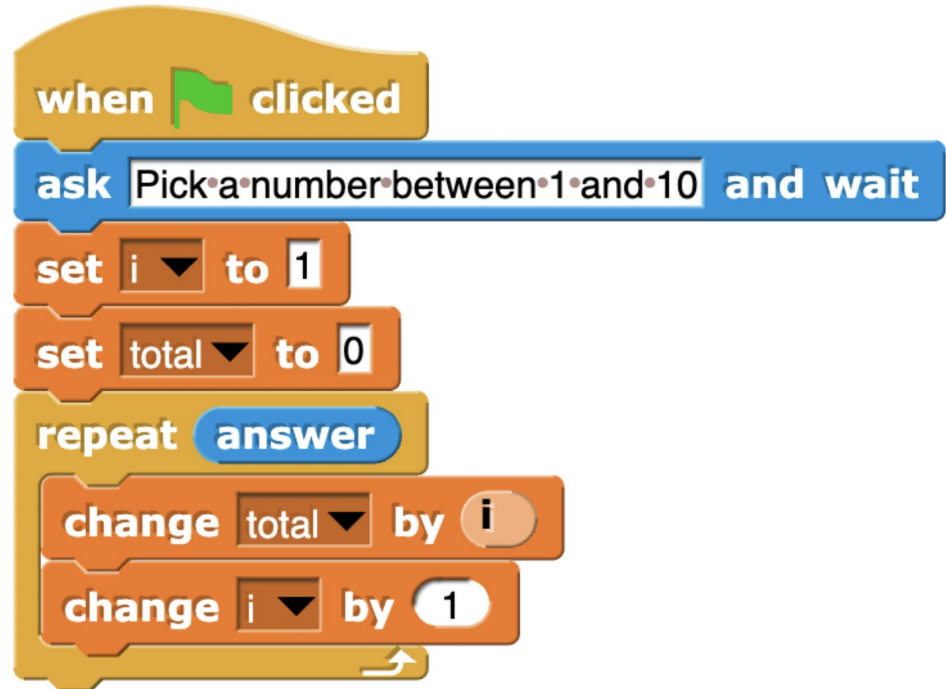
A. 1

B. 2

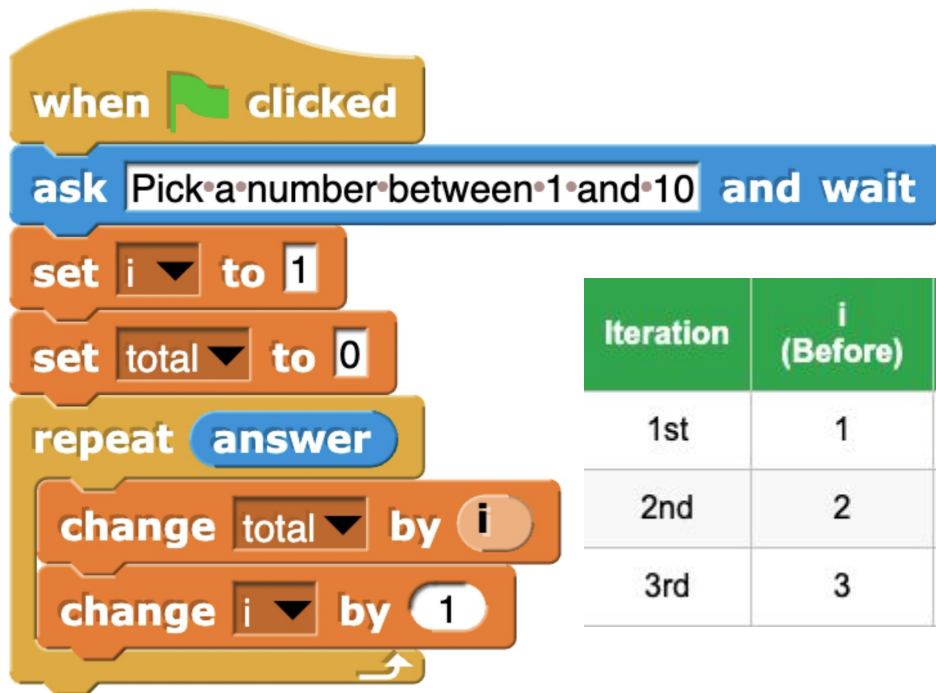
C. 3

D. 4

E. 5



Q: What is the value in **i** when the code is run, assuming **user input = 3**?



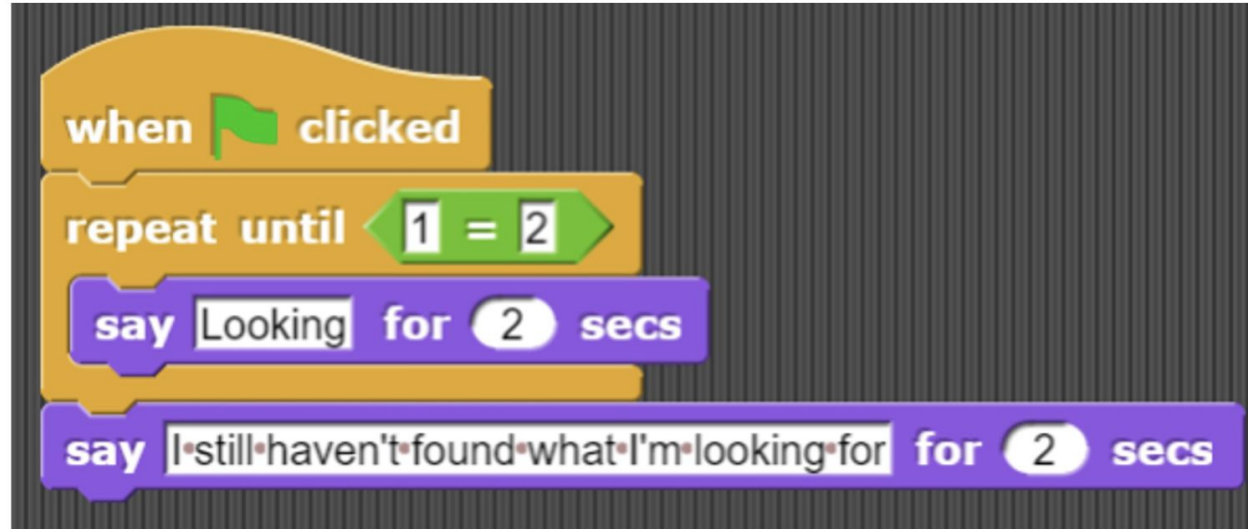
Iteration	i (Before)	total (Before)	Add i to total	total (After)	i +1	i (After)
1st	1	0	$0 + 1$	1	+1	2
2nd	2	1	$1 + 2$	3	+1	3
3rd	3	3	$3 + 3$	6	+1	4



**Q: Will this program ever say “I still haven’t found what I’m looking for”?**



- A. Yes
- B. No
- C. Sometimes



# For loops

when  clicked

for **i** = 1 to 10

say **i** for 1 secs

say Bye-bye! for 2 secs

when  clicked

for  = 1 to 10

say  for 1 secs

say Bye-bye! for 2 secs

**i is a variable**



when  clicked

for  = 1 to 10

say  for 1 secs

say Bye-bye! for 2 secs

**i is initialized  
with a value of 1**

when  clicked

for **i** = 1 to 10

say **i** for 1 secs

say Bye-bye! for 2 secs

**This loop will run until i has a value that is not between 1 to 10 (inclusive)**

when  clicked

for **i** = 1 to 10

say **i** for 1 secs

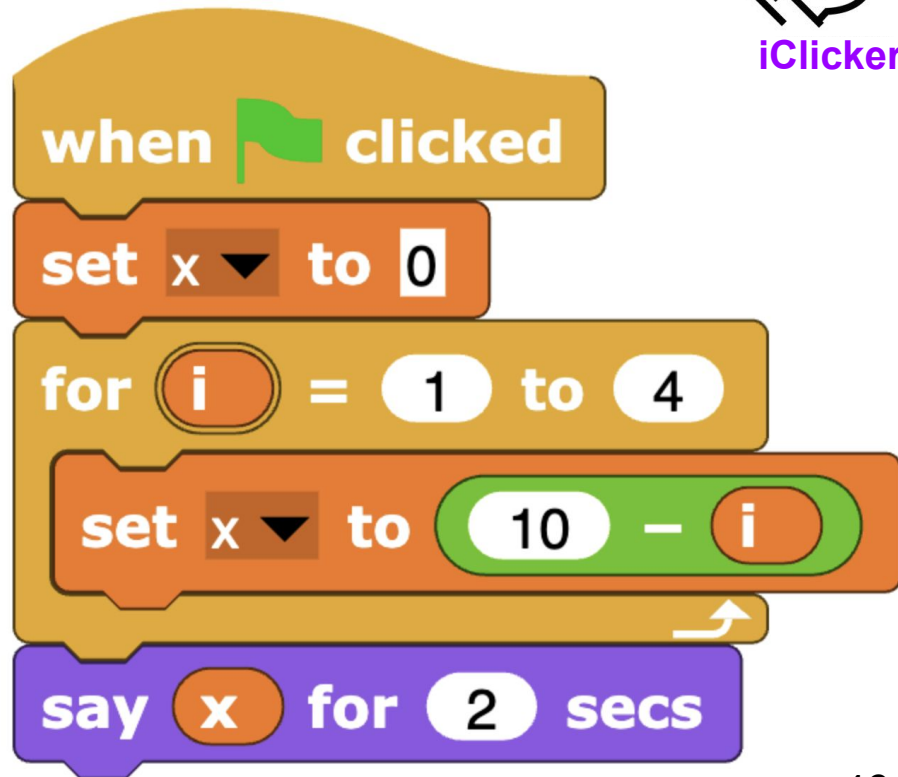
say Bye-bye! for 2 secs

**Every time we reach the end of the loop, i will increase (increment) by 1**

Q: What is the value in **x** when the code is run?



- A. 1
- B. 3
- C. 4
- D. 6
- E. 10



# That's it!

That's all the  
programming  
basics you need to  
know *...(for now).*

Programming is a LOT  
easier to learn by  
*doing* than by  
*watching!*

# Take-Home Activity



**Q: There's no  $\leq$  block in Snap! Suppose we wanted to build one. Which of the following Boolean expressions is equivalent to the expression  ?**



iClicker

A



B



C



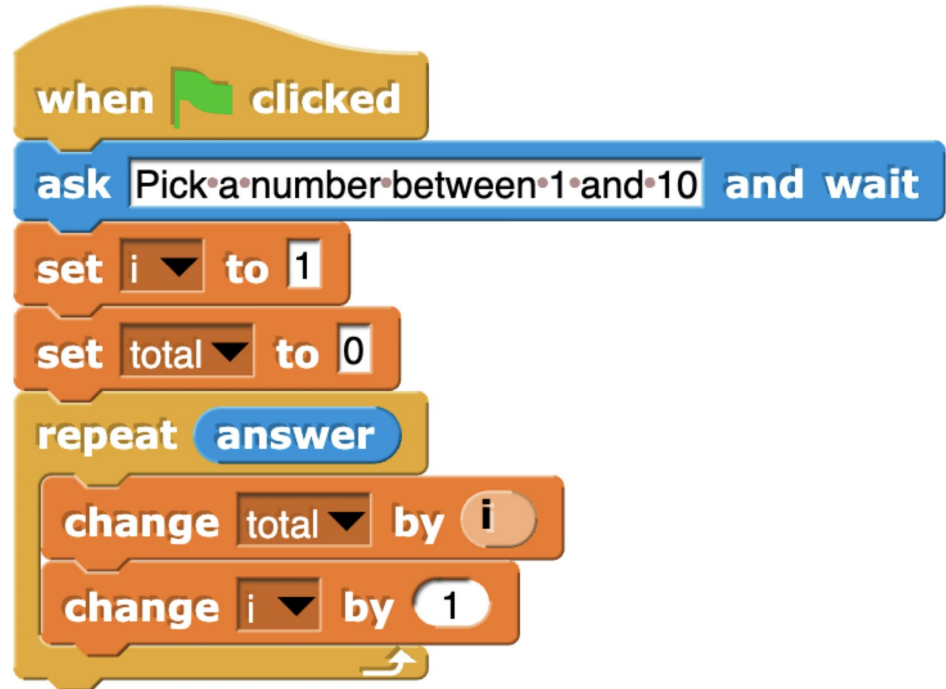
D



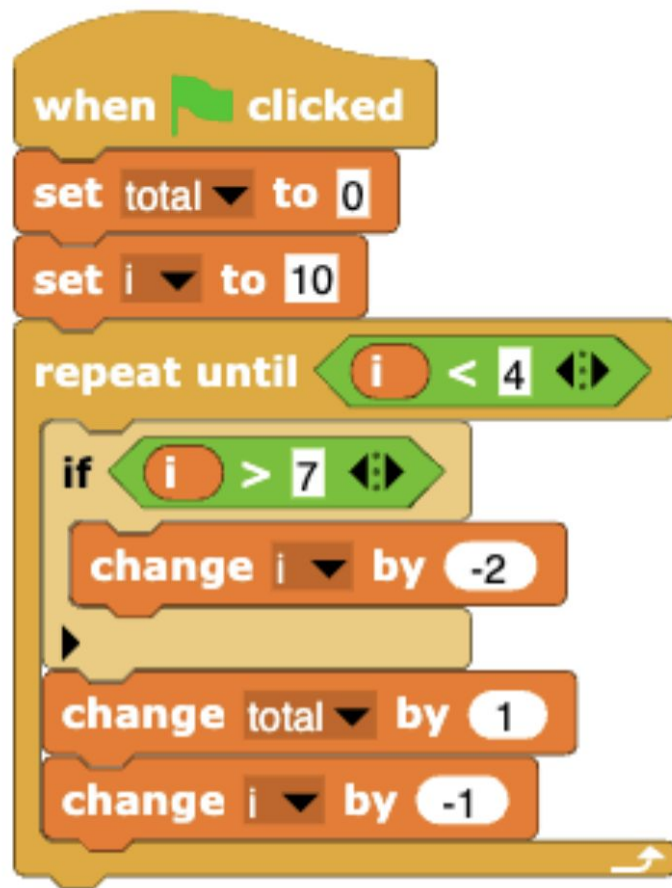
**Q: What is the value in **total** when the code is run, assuming **user input = 3**?**



- A. 2
- B. 3
- C. 4
- D. 6
- E. 10

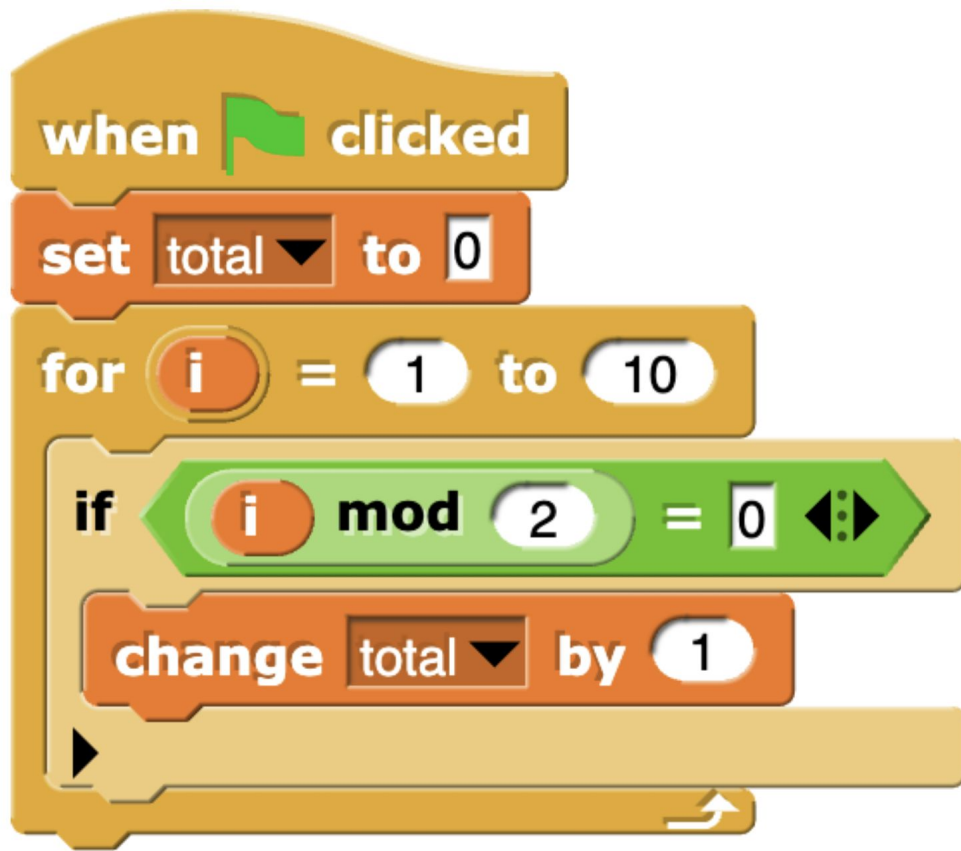


Q: What is the value of **total** and **i** when this code block is run?



**Q: What is the value of **total** when this code block is run?**

**What does this code block do?**



# Wrap up



# Wrap Up

- **Lab #4**
  - The Evolution of Trust; <https://ncase.me/trust/>
  - Review/Play Game **before your lab**
  - Due on Thursday, Feb 6 at 11:59pm
- **Post-Class (PC) Quiz #3**
  - Only 1 attempt, 60 minutes
  - To be posted tonight; Due on Sunday, Feb 9 at 11:59pm
- **Midterm**
  - All content covered from Week 1 Until end of Week 5 (this Friday)
  - We will likely be writing in a **different room**; I will confirm soon
  - Past Exams/Practice Questions to be released soon