Lecture-4

Expression of algorithms

- Pseudocode
  - Going from algorithm to code
- Flow charts

Going from algorithm to code

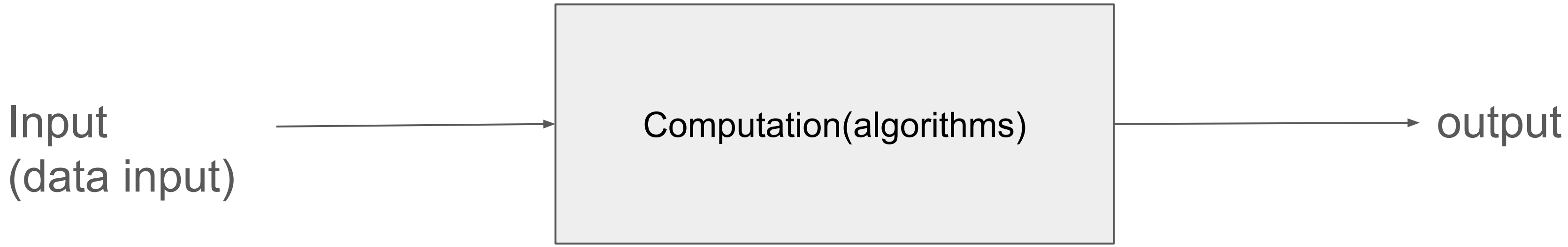
- Scratch, c, python examples

Last week: CPU

- ALU,
- CU
  - Instruction set (in ASM, MOV, DIV, ADD, etc)
-

Last week: Program

- Written set of instructions
  - Takes input
  - Applies a set of instructions
  - And output
- Computations are based on some algorithms



Last week: algorithm

Set of steps to solve a problem

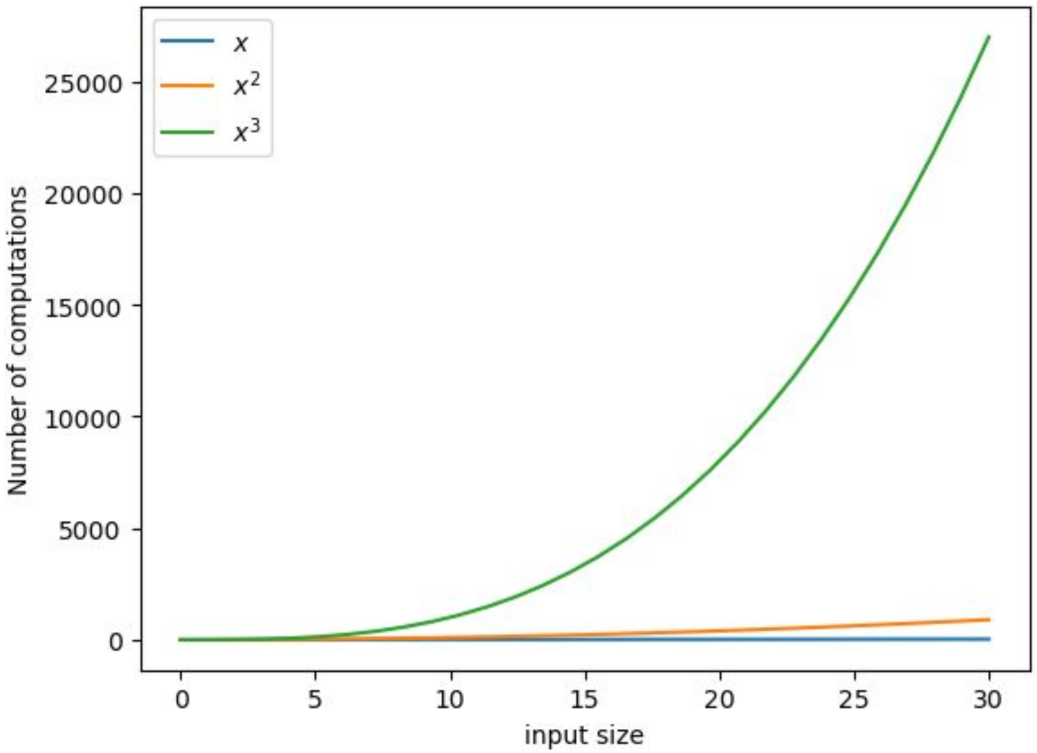
- Steps are specific
- Non-ambiguous

Last week: Number of computations

Number of arithmetic and logic operations

Number of steps

Number of repetitions

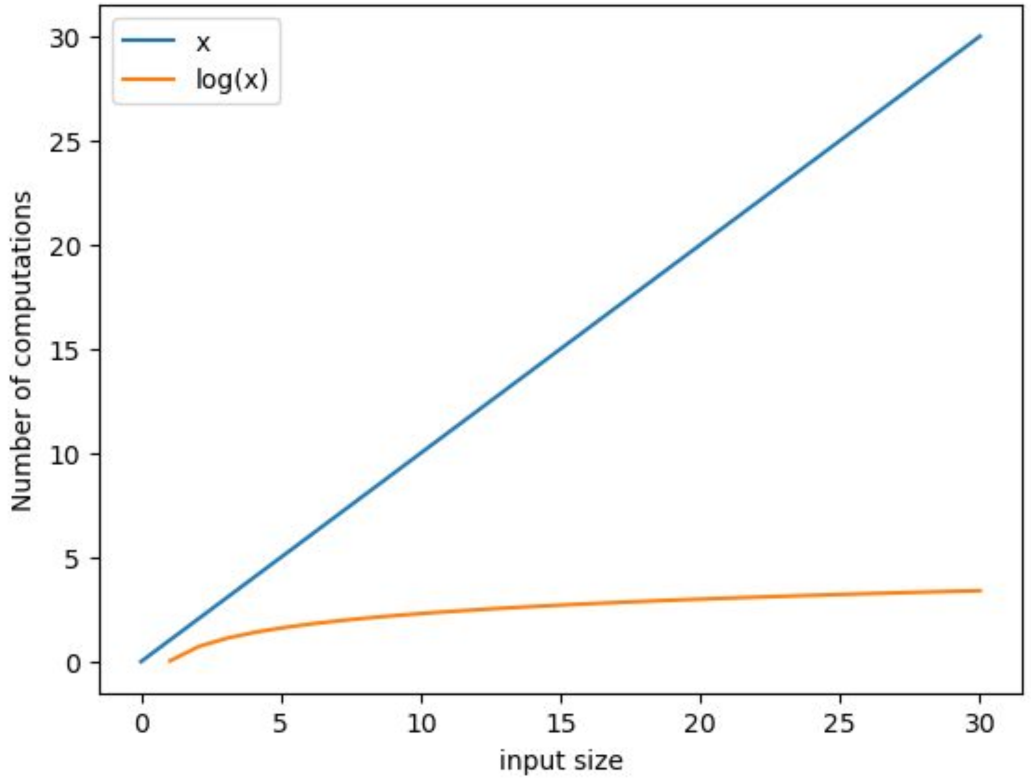


Last week: Number of computations

Number of arithmetic and logic operations

Number of steps

Number of repetitions

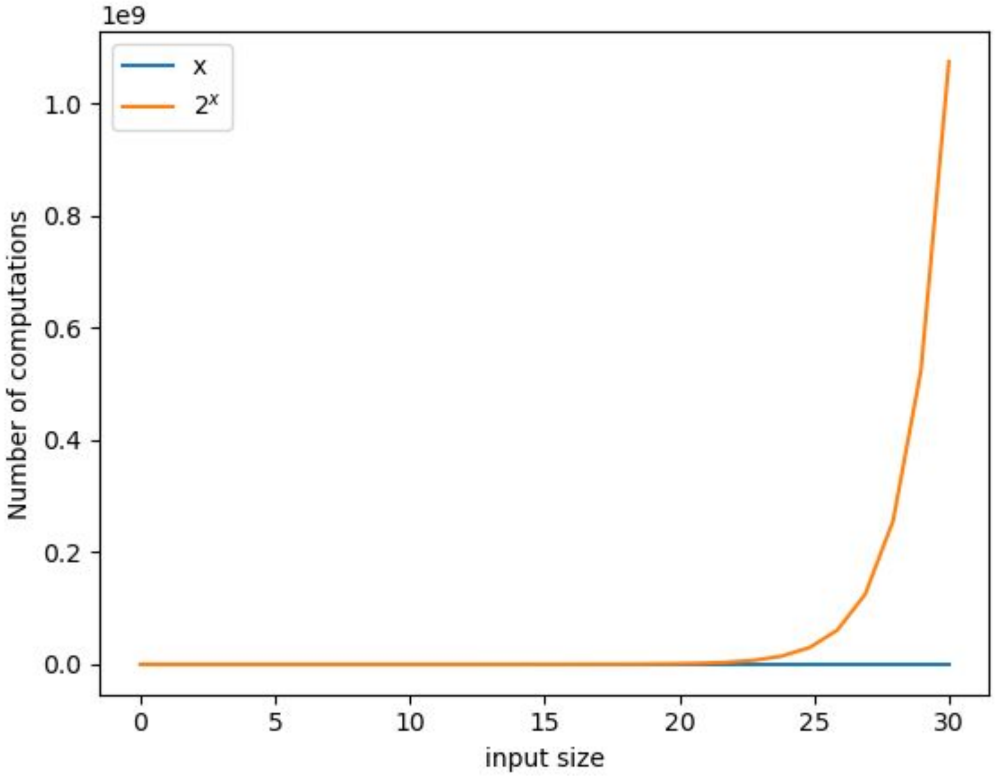


Last week: Number of computations

Number of arithmetic and logic operations

Number of steps

Number of repetitions



Example: searching a person in campus

Write an algorithm for finding/searching a person

- Random searching
    - Randomly choose a person and randomly move another direction and choose another person, do this until you find the person.

How to express this in a more elegant way?

- We should be able to easily write code by following the steps

Pseudocode:
Description of an algorithm in a code like structure using plain math and text.

- Neither code nor english

Example: searching a person in campus

- Random searching
    - Input:
        - **x**: the name of the person being searched
    - Repeat the following steps
        - Randomly choose a location
        - Randomly choose a person in that location
        - If it is x
            - Done!

Random searching-2

- Input:
    - **x**: the name of the person being searched
- Repeat the following steps
    - Randomly choose a location, L
    - Randomly choose a person **y** in L
    - **If x == y**
        - Done!
    - **Else**
        - Not done!

Example: searching a person in campus

Random searching-2

- Input:
  - **x**: the name of the person being searched
- While not done
  - L = random_location()
  - **y** = random_person(L)
  - **If compare(x, y)**
    - Done!
  - **Else**
    - Not done!

- random_location()
  - Input
  - Output
    - L: A random location
  - Algorithm steps…
- random_person(L)
  - Input
    - L: a location
  - Output
    - y: a random person in location L
  - …

```
1   Pick up phone book
2   Open to middle of phone book
3   Look at page
4   If Smith is on page
5       Call Mike
6   Else if Smith is earlier in book
7       Open to middle of left half of book
8       Go back to line 3
9   Else if Smith is later in book
10      Open to middle of right half of book
11      Go back to line 3
12  Else
13      Quit
```

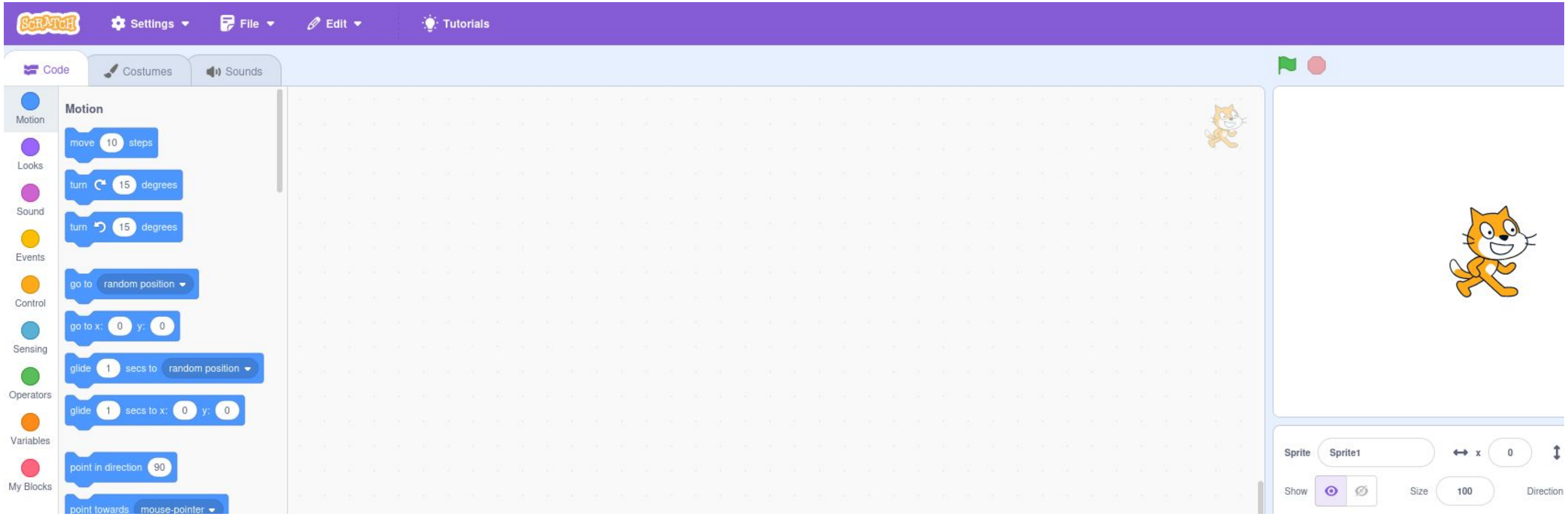
```
1    Pick up phone book
2    Open to middle of phone book
3    Look at page
4    If Smith is on page
5        Call Mike
6    Else if Smith is earlier in book
7        Open to middle of left half of book
8        Go back to line 3
9    Else if Smith is later in book
10       Open to middle of right half of book
11       Go back to line 3
12   Else
13       Quit
```

```
1    Pick up phone book
2    Open to middle of phone book
3    Look at page
4    If Smith is on page
5        Call Mike
6    Else if Smith is earlier in book
7        Open to middle of left half of book
8        Go back to line 3
9    Else if Smith is later in book
10       Open to middle of right half of book
11       Go back to line 3
12   Else
13       Quit
```

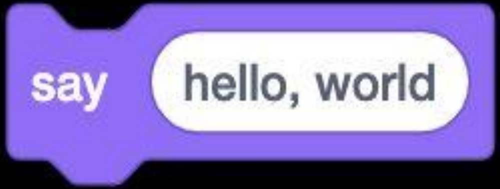
```
1    Pick up phone book
2    Open to middle of phone book
3    Look at page
4    If Smith is on page
5        Call Mike
6    Else if Smith is earlier in book
7        Open to middle of left half of book
8        Go back to line 3
9    Else if Smith is later in book
10       Open to middle of right half of book
11       Go back to line 3
12   Else
13       Quit
```

```
1    Pick up phone book
2    Open to middle of phone book
3    Look at page
4    If Smith is on page
5        Call Mike
6    Else if Smith is earlier in book
7        Open to middle of left half of book
8        Go back to line 3
9    Else if Smith is later in book
10       Open to middle of right half of book
11       Go back to line 3
12   Else
13       Quit
```

- functions
- conditions
- Boolean expressions
- loops

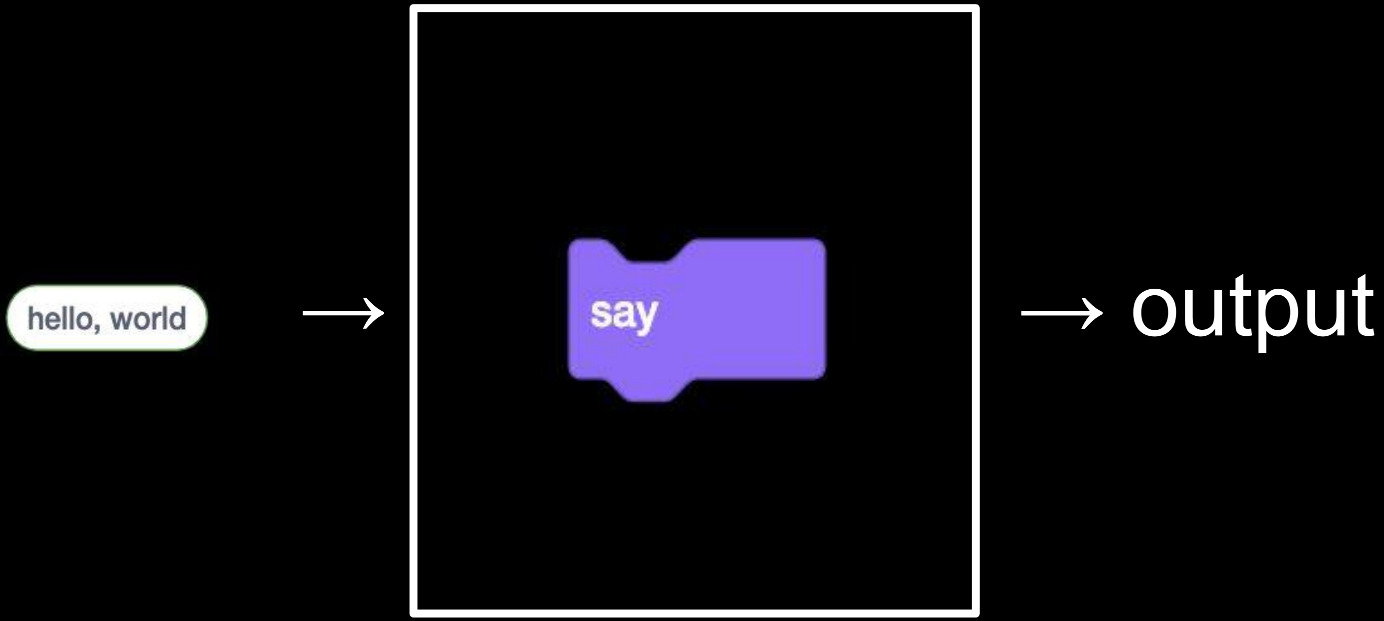


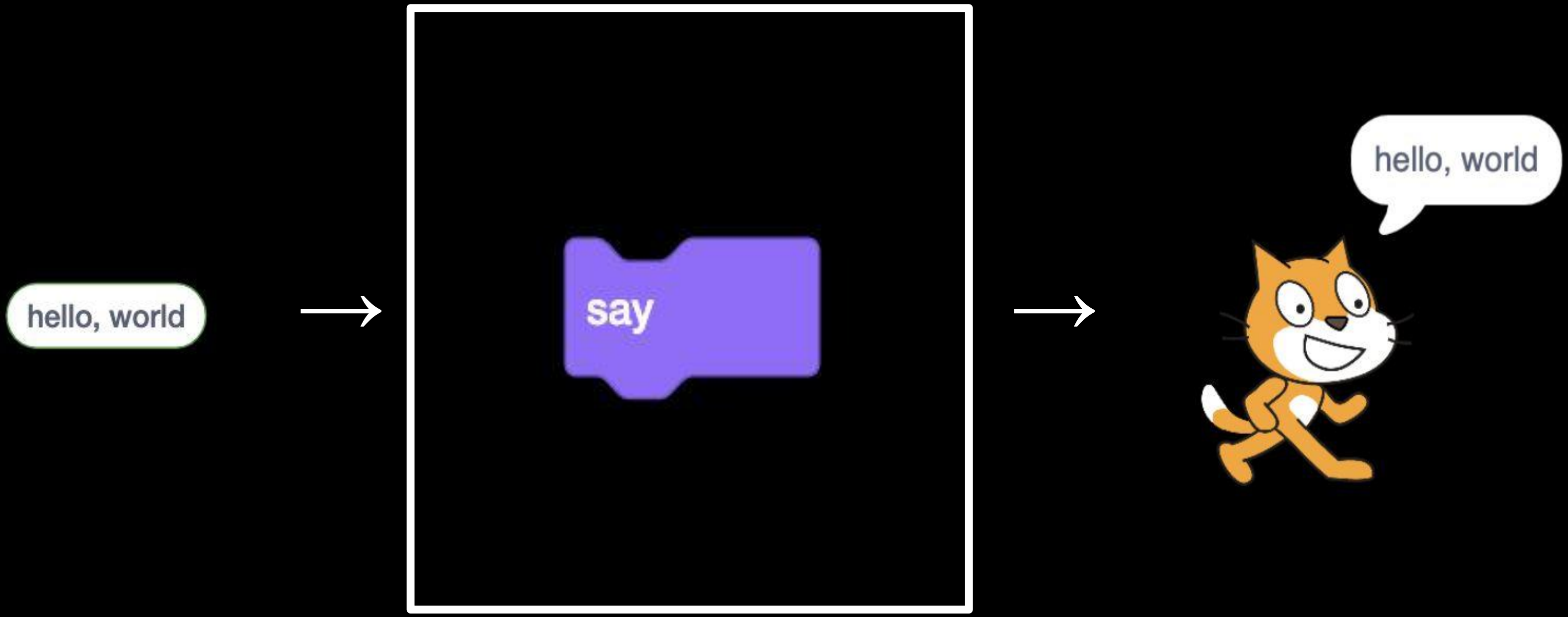
Scratch
https://scratch.mit.edu
a coding language with a simple visual interface

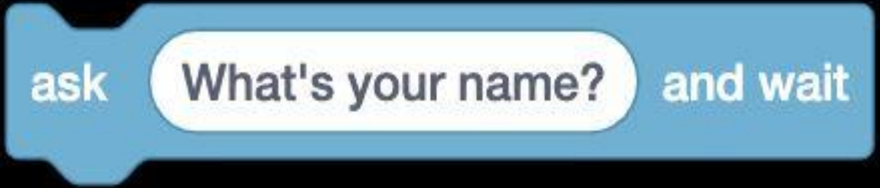

say hello, world

$$\text{input} \rightarrow \boxed{\text{algorithms}} \rightarrow \text{output}$$

$$\begin{tikzcd}\text{hello, world} \arrow[r] & \boxed{\text{algorithms}} \arrow[r] & \text{output}\end{tikzcd}$$


hello, world
say
output


hello, world
say
hello, world

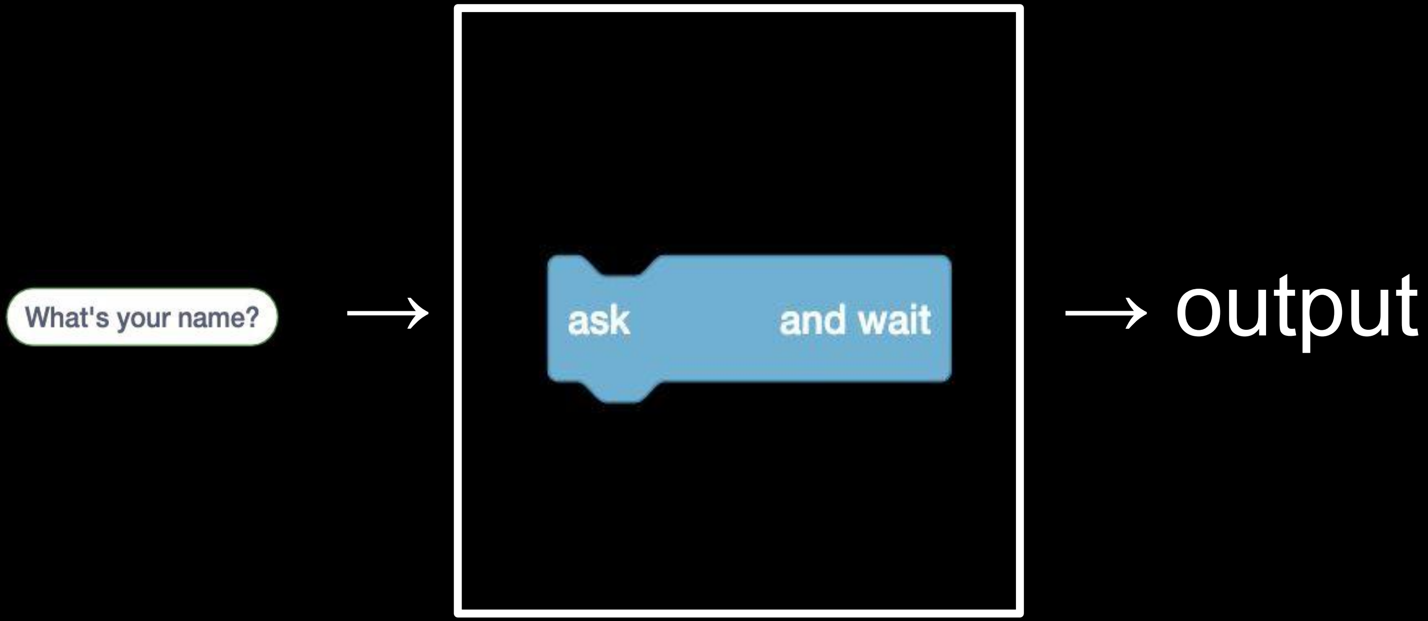


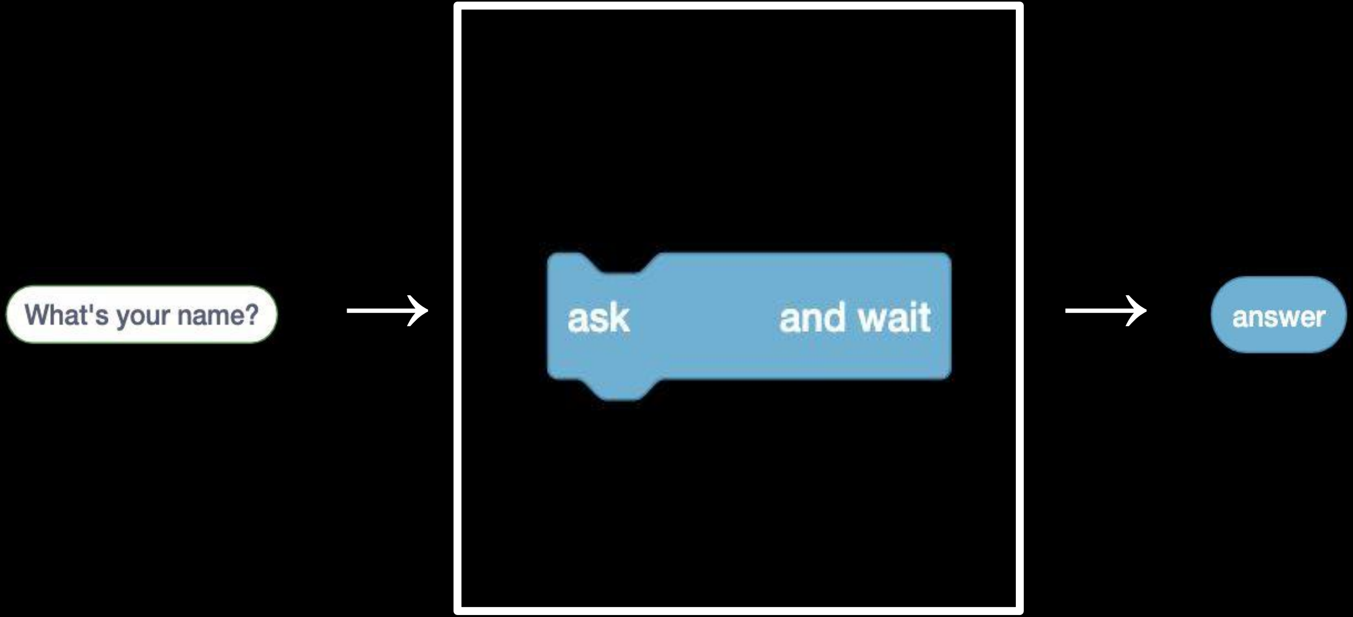
```
ask (What's your name?) and wait
```

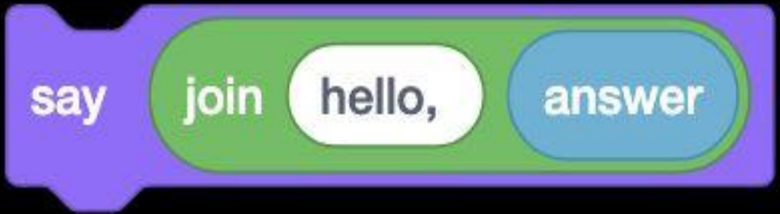
$$\begin{tikzcd} \text{input} \arrow[r] & \boxed{\text{algorithms}} \arrow[r] & \text{output} \end{tikzcd}$$

What's your name?

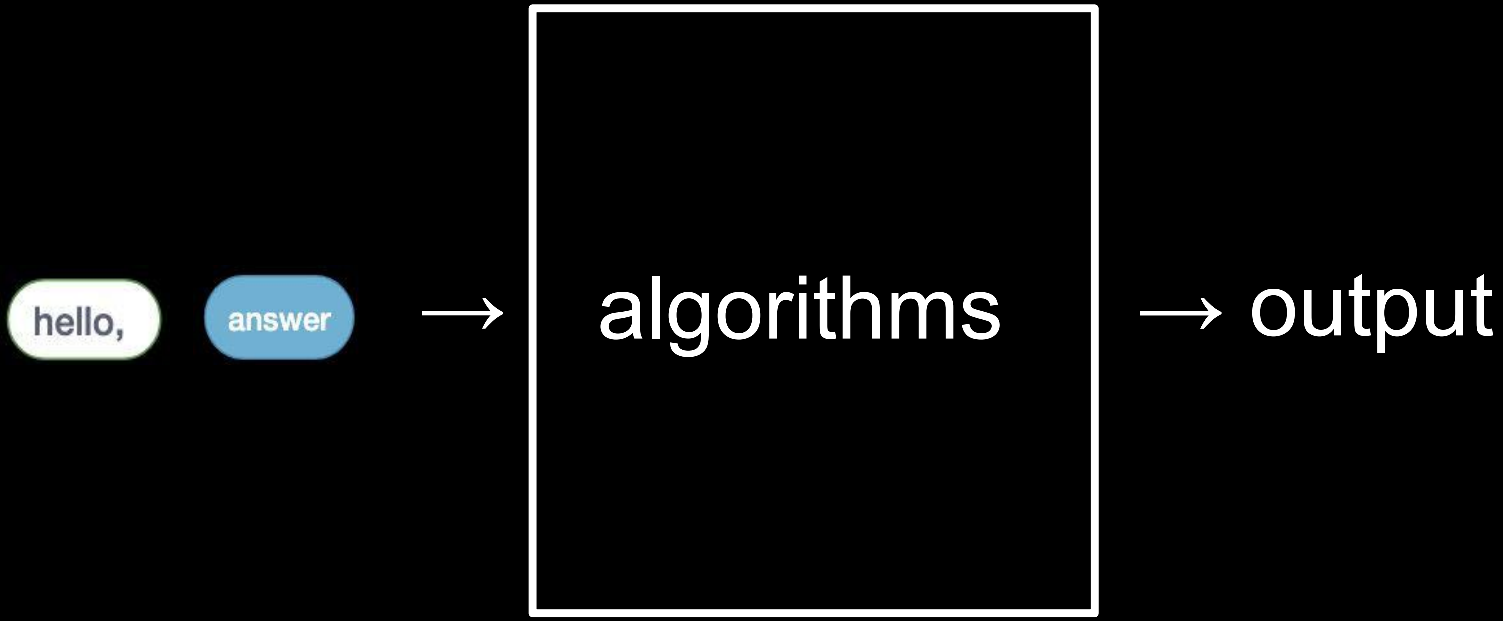
$\rightarrow \boxed{\text{algorithms}} \rightarrow$ output

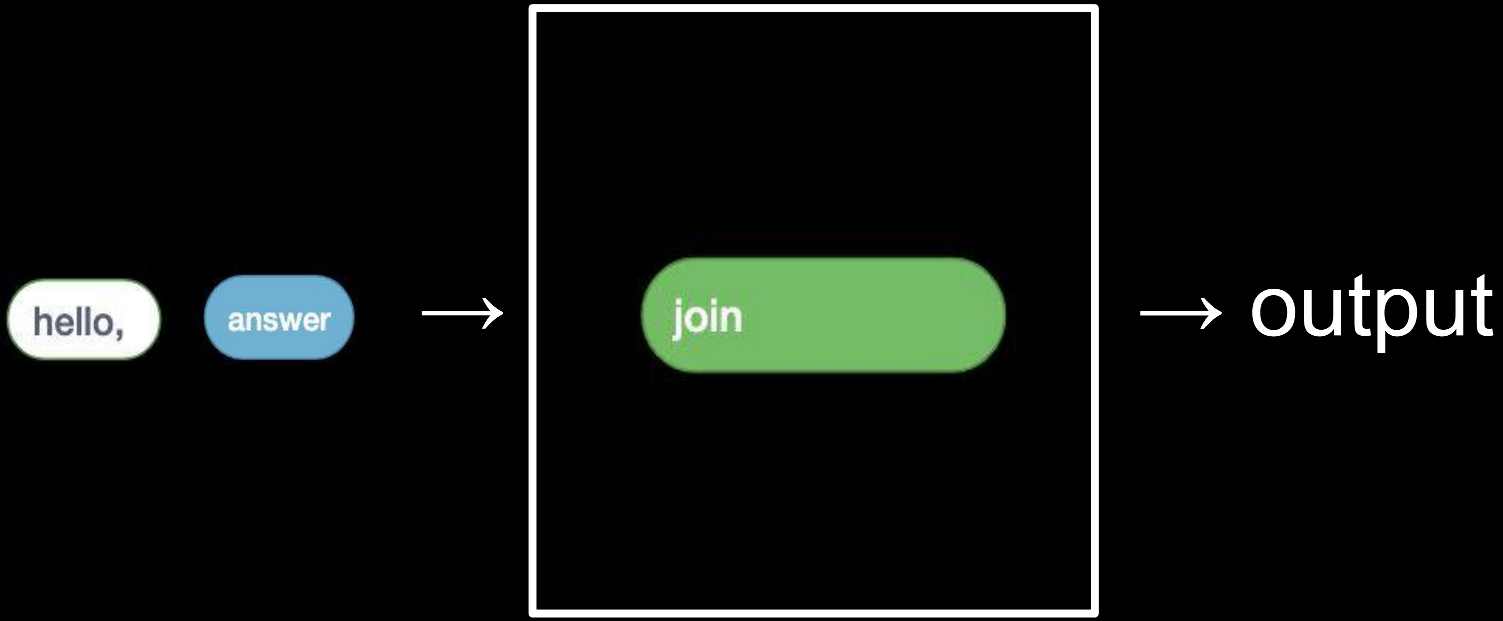

What's your name?
→
ask and wait
→ output

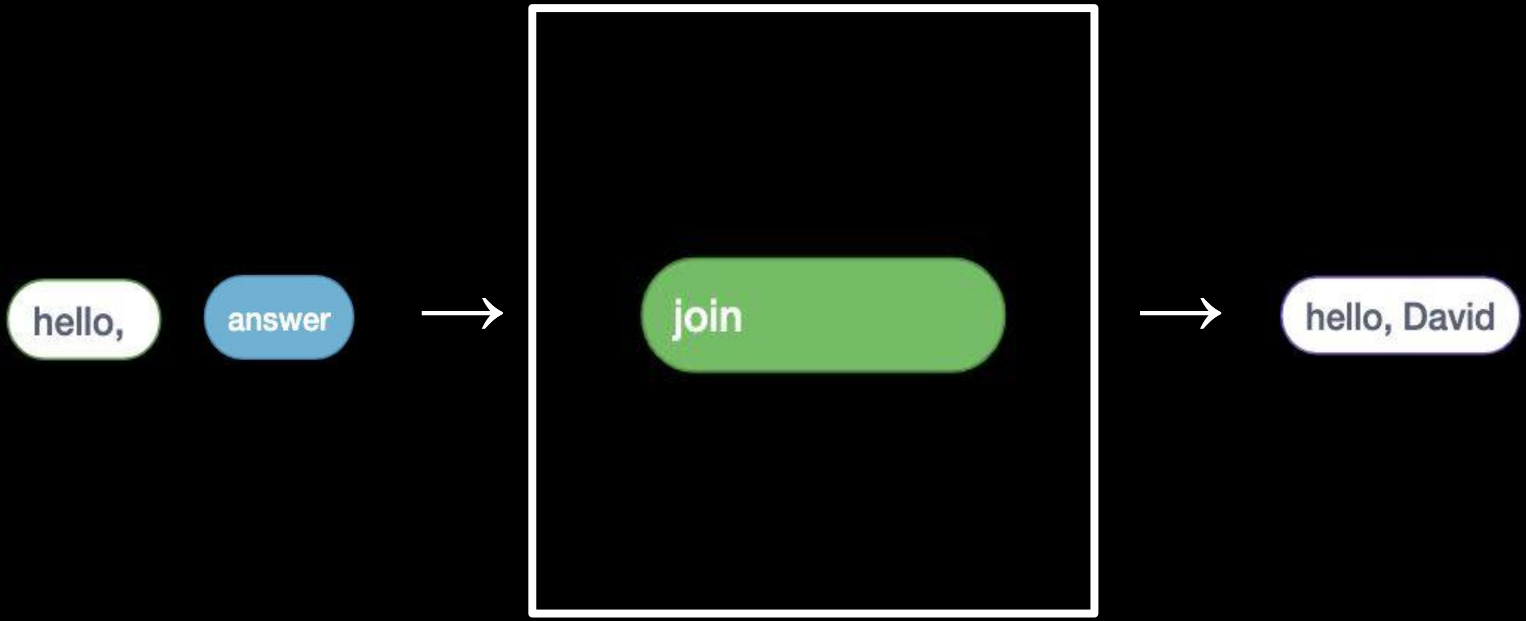

What's your name?
ask and wait
answer


say
join
hello,
answer

$$\text{input} \rightarrow \boxed{\text{algorithms}} \rightarrow \text{output}$$

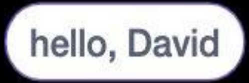

hello,
answer
→
algorithms
→ output


hello,
answer
join
output

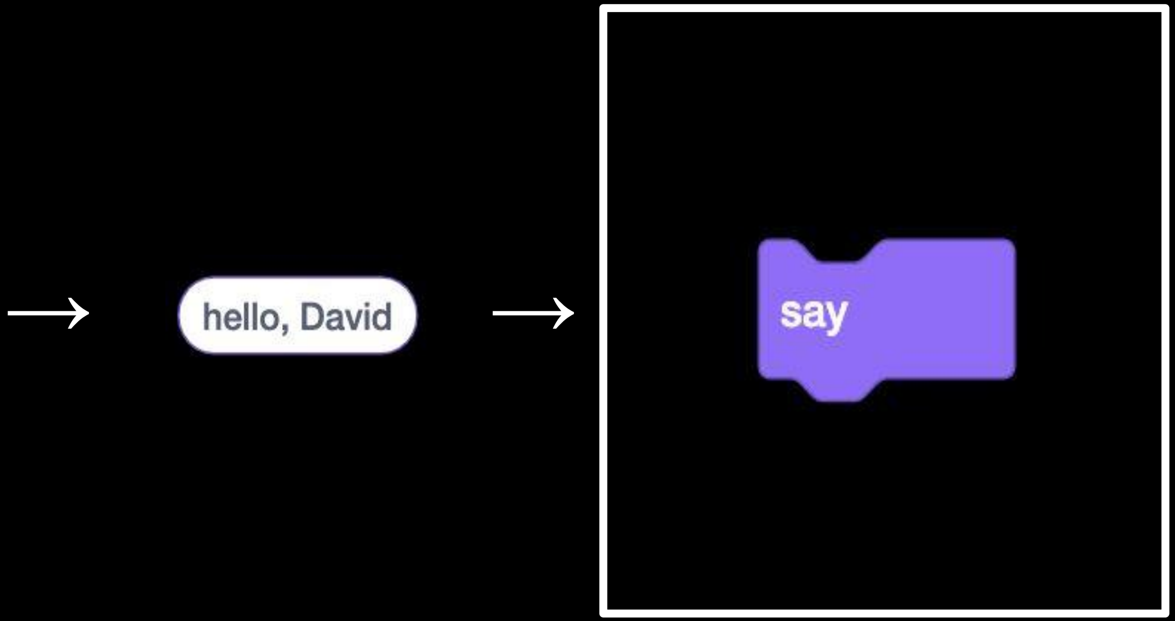

hello,
answer
join
hello, David

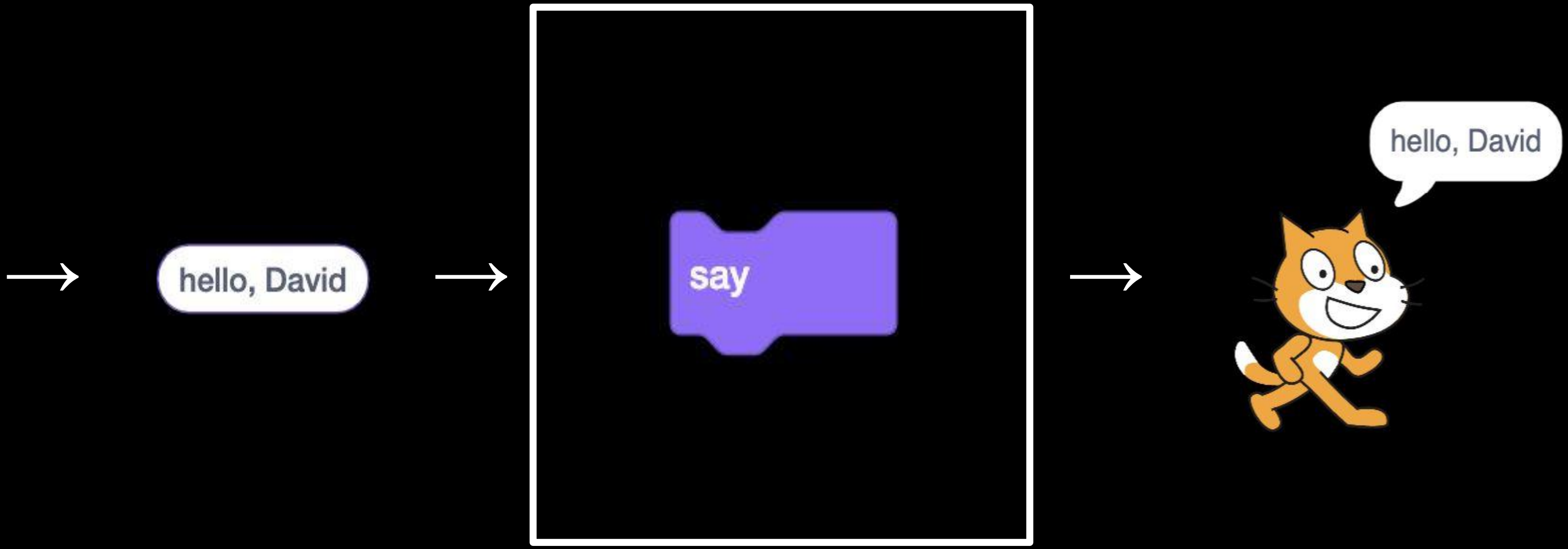
$\longrightarrow$ hello, David

→



hello, David

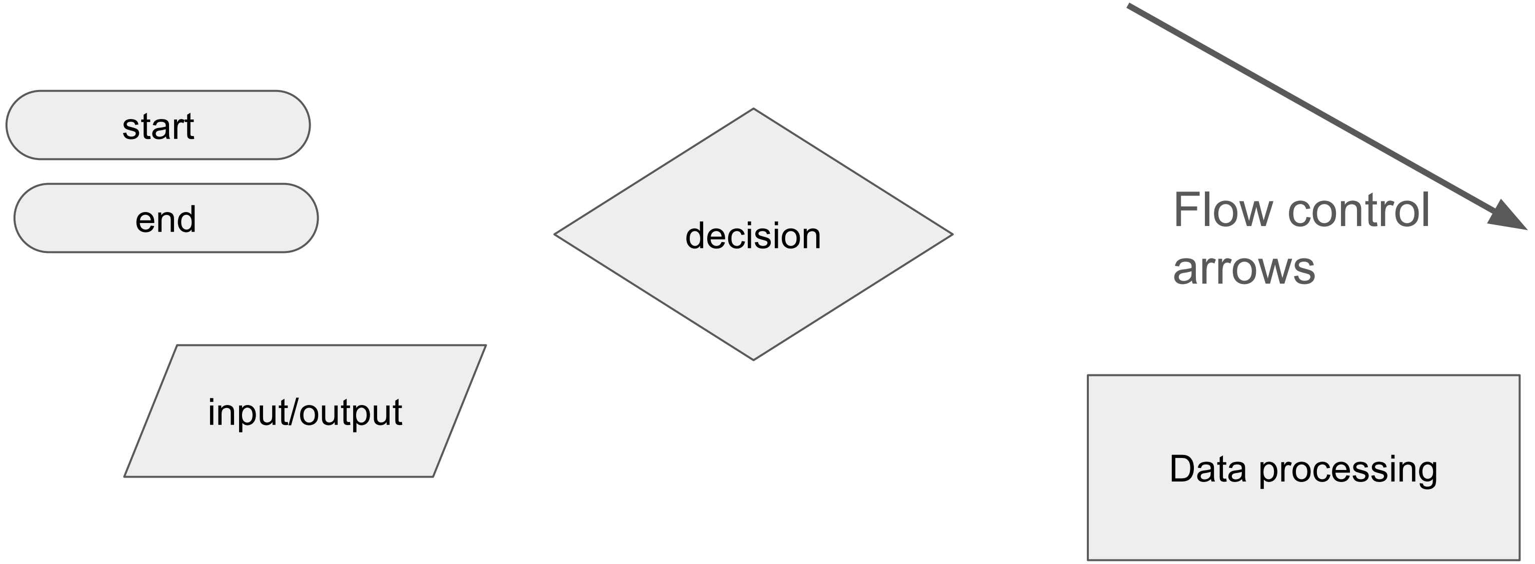

hello, David
say


hello, David
say
hello, David

Example: compute hourly payment

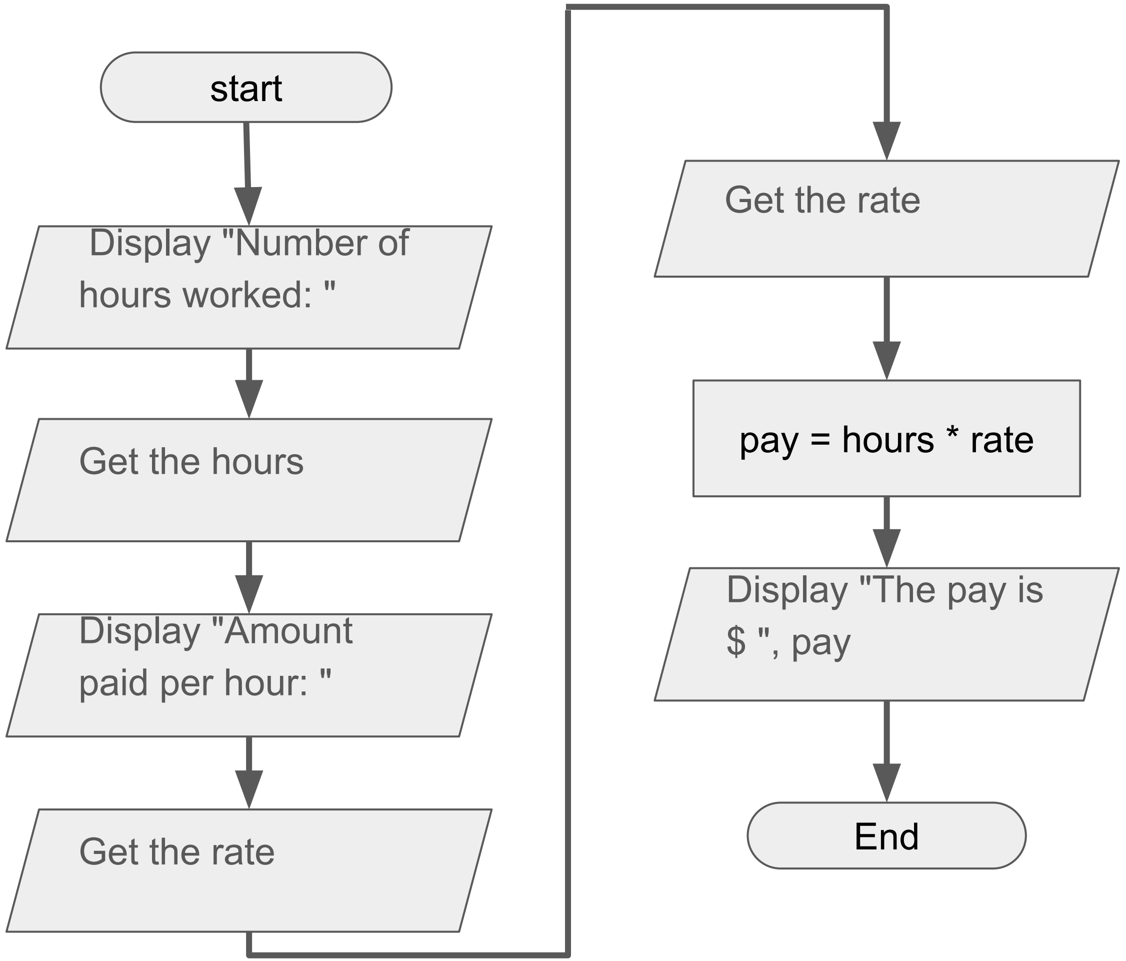
```
1. Display "Number of hours worked: "
2. Get the hours
3. Display "Amount paid per hour: "
4. Get the rate
5. Compute pay = hours * rate
6. Display "The pay is $" , pay
```

Flowchart Symbols



Example: compute hourly payment

1. Display "Number of hours worked: "
2. Get the hours
3. Display "Amount paid per hour: "
4. Get the rate
5. Compute pay = hours * rate
6. Display "The pay is $" , pay


start
Display "Number of hours worked: "
Get the hours
Display "Amount paid per hour: "
Get the rate
Get the rate
pay = hours * rate
Display "The pay is $ ", pay
End

Write scratch code

C and Python code

```c
/*C code*/
#include <stdio.h>
int main(){
   float hours, rate, pay;
   printf("Number of hours worked:
");
   scanf("%f", &hours);
   printf("Amount paid per hour: ")
   scanf("%f", &rate);
   pay = hours * rate;
   printf("The pay is $%f", pay);
}
```

```python
#python code
print("Number of hours worked: ")
hours = float(input('the hours'))
print("Amount paid per hour: ")
rate  = float(input('the rate'))
pay = hours * rate
print("The pay is $" , pay)
```

Example: An app to monitor pulse rate of a patient

Pulse rate, also known as your heart rate, is the number of times the heart beats per minute.

A normal resting heart rate should be between 60 to 100 beats per minute, but it can vary from minute to minute.

Design an algorithm that monitor a heart rate of patient and contacts the healthcare professional when it is abnormal.

Example: An app to monitor pulse rate of a patient

Monitor the pulse rate of a patient

- Let’s assume every 5 seconds.

If the pulse rate is **between 60 and 100** then display “Normal”.

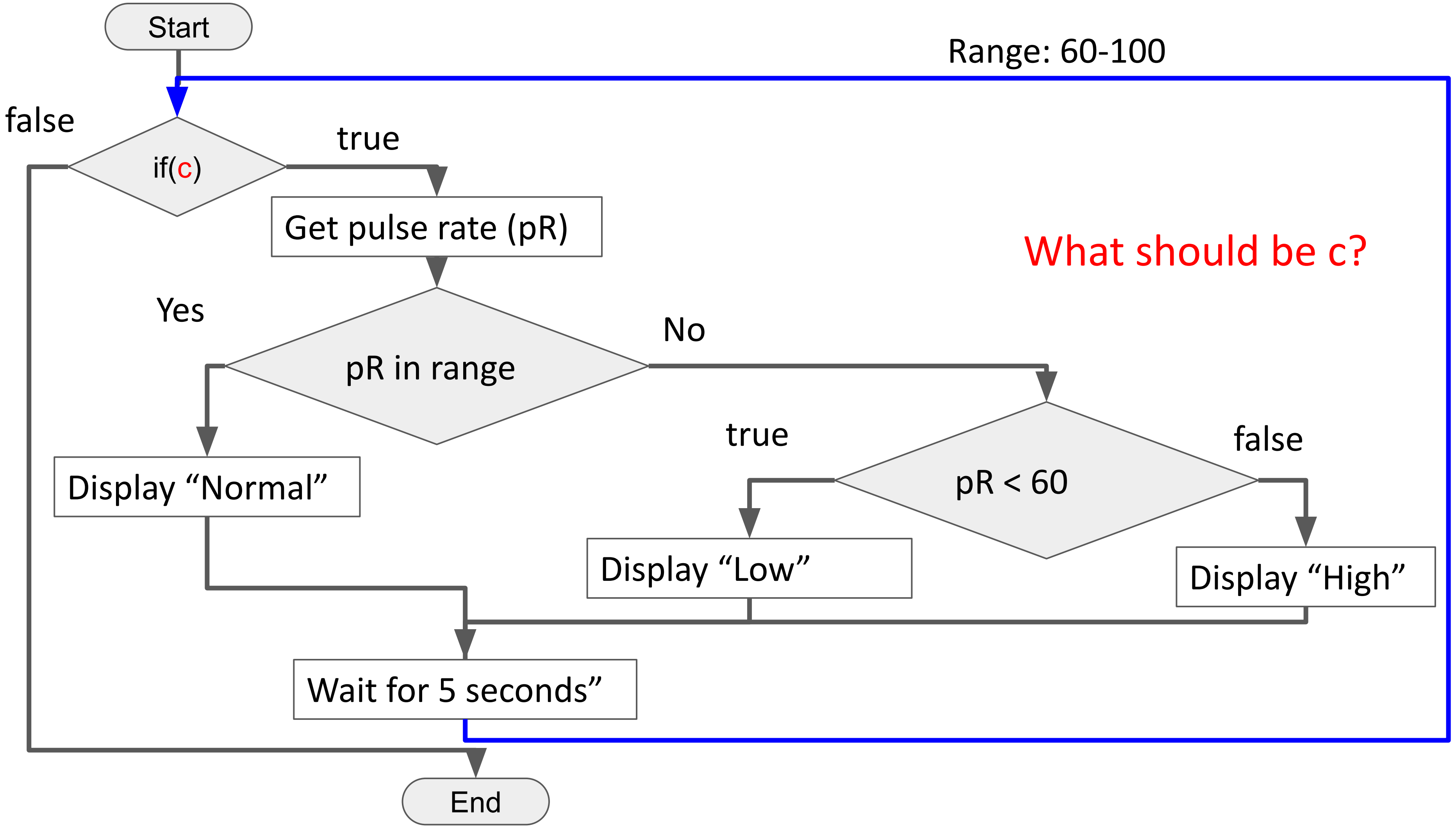
If the rate is below 60, then display “Low”.

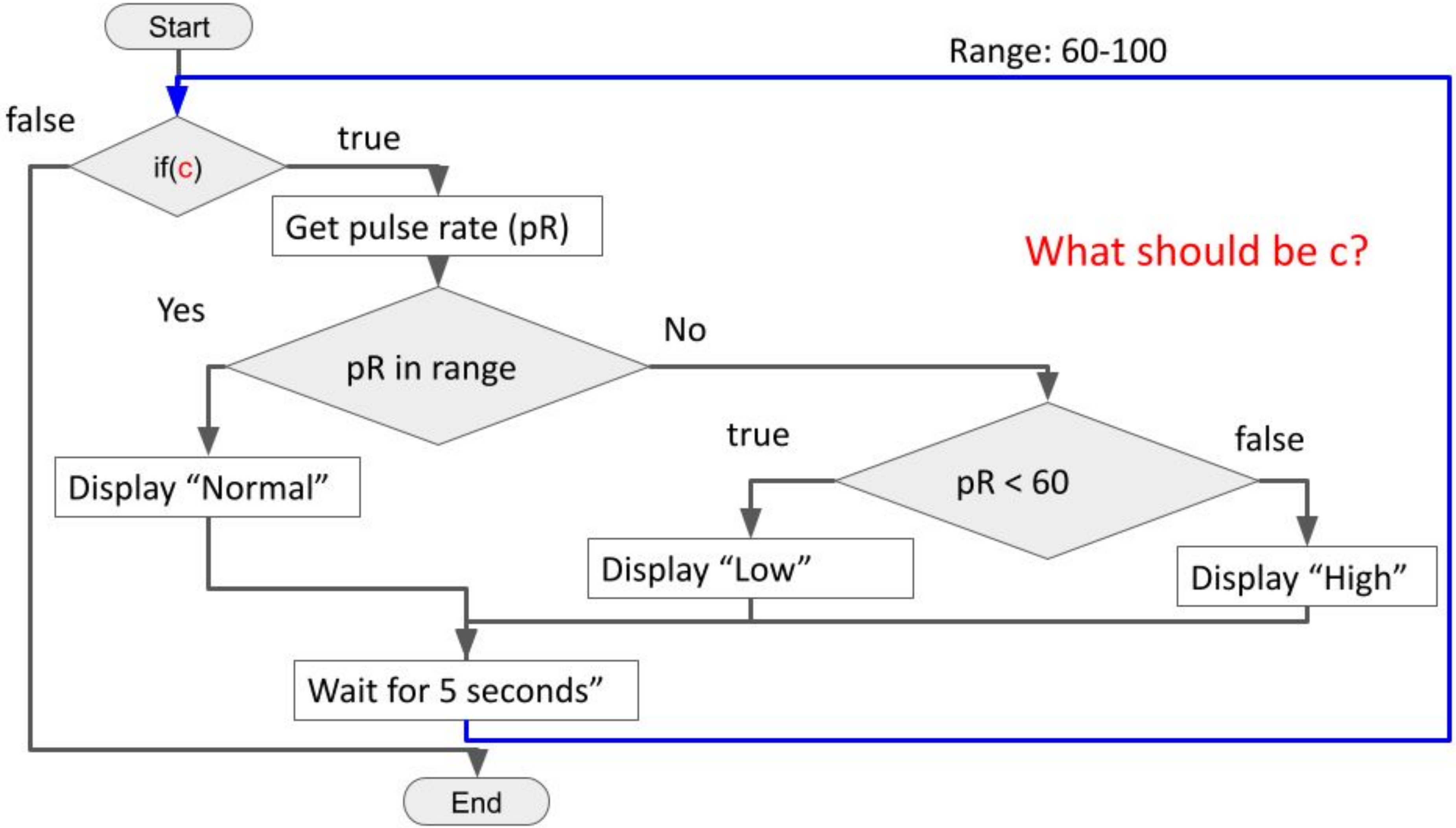
If the rate is above 100 then display “High”.

Pseudocode: Algorithm on Text

- Input
    - Pulse rate
- Output
    - Normal
    - Low
    - High

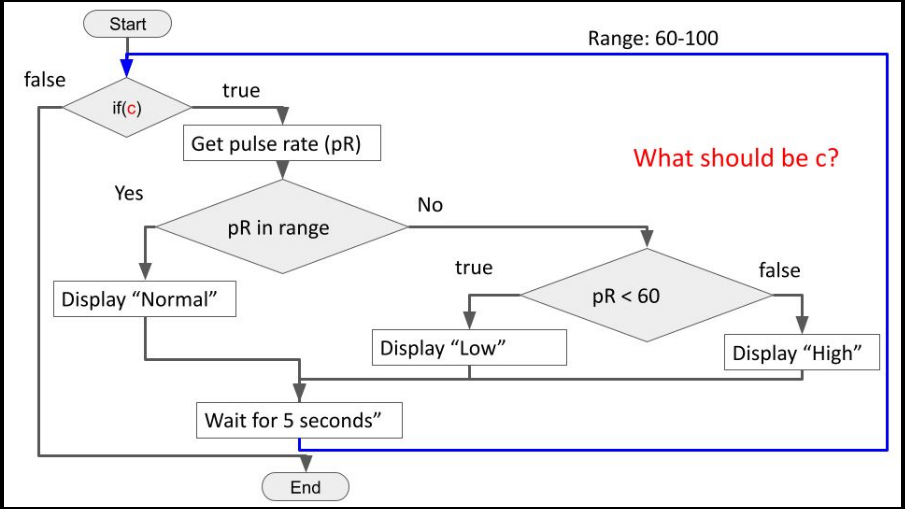
```
Repeat forever
    get pulse rate;
    if the pulse rate is between 60 and 100 then
        Display "Normal";
    else
        if the pulse rate is<60 then
            Display "Low";
        else
            Display "High";
    Wait for 5 seconds;
```

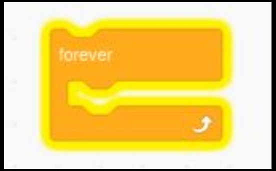

Start
Range: 60-100
false
if(c)
true
Get pulse rate (pR)
What should be c?
Yes
No
pR in range
true
false
pR < 60
Display "Normal"
Display "Low"
Display "High"
Wait for 5 seconds"
End

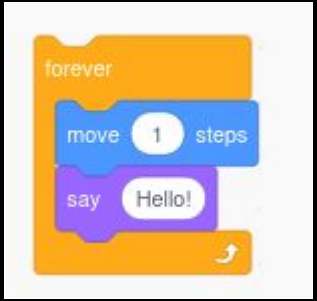


What should be c?

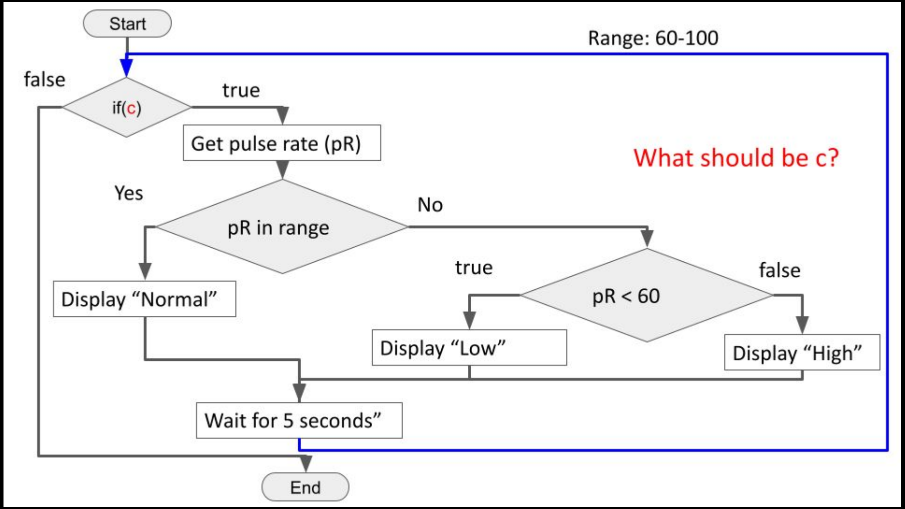
Scratch implementation

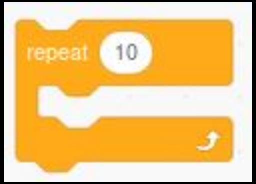


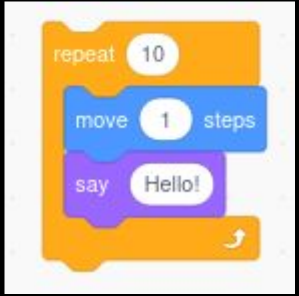


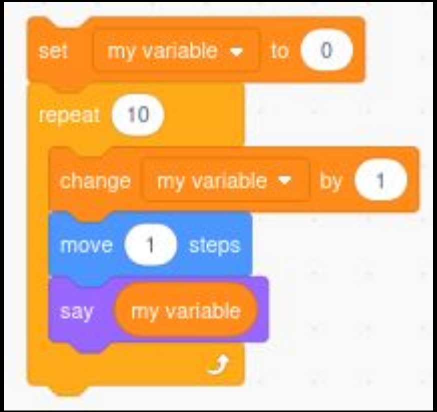


Scratch implementation

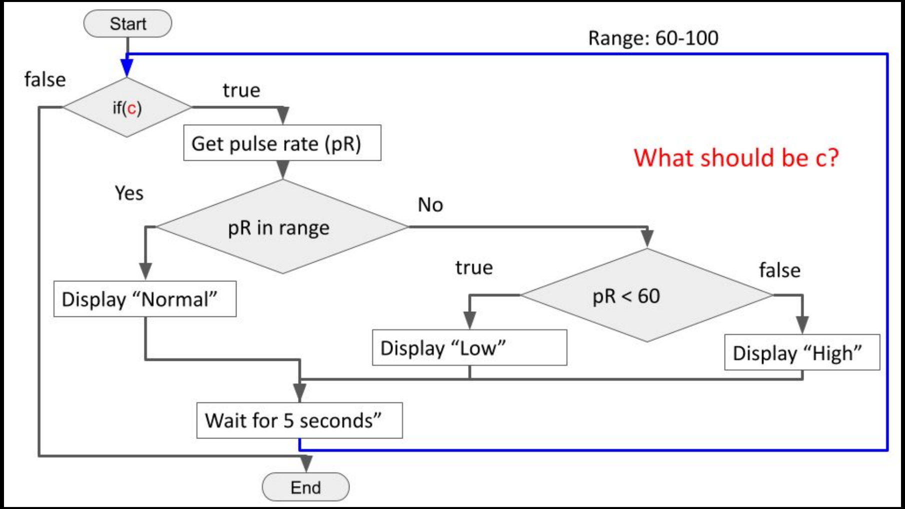


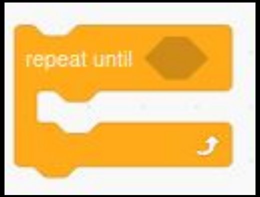




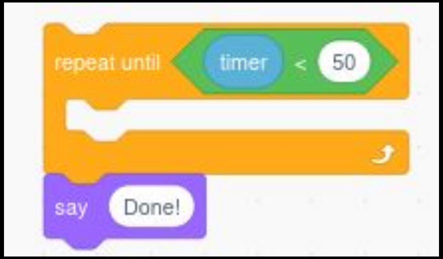


Scratch implementation

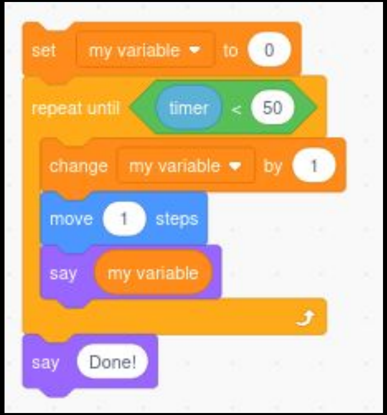




```
repeat until < >
↺
```



```
repeat until <(timer) < (50)>
↺
say (Done!)
```



```
set [my variable ▾] to (0)
repeat until <(timer) < (50)>
    change [my variable ▾] by (1)
    move (1) steps
    say (my variable)
↺
say (Done!)
```

*/*C implementation*/*

```c
#include <stdio.h>

int main(){
   float pR;
   while(c):
       scanf("%f", &pR);
       if(pR >= 60 && pR <= 100){
         /*normal*/
           printf("normal");
       }
       else { /*not normal*/
           if(pR < 60){
               printf("low");
           else{
               printf("high");
           }
       }
       sleep(1);
   }
}
```

#Python implementation

```python
import time
c = True
while(c):
   pR = float(input("pR: "))
   if (pR >= 60 and pR <= 100):#normal
       print("normal")
   else: #not normal
       if(pR < 60):
           print("low");
       else:
           print("high");
   time.sleep(1)
```

Next week

How to run code on computers

How to manage multiple programs

- Computer operating systems
  - Unix, Linux, MacOS, Windows, iOS,Android
  - Linux installation
  - Linux terminal commands
  - …