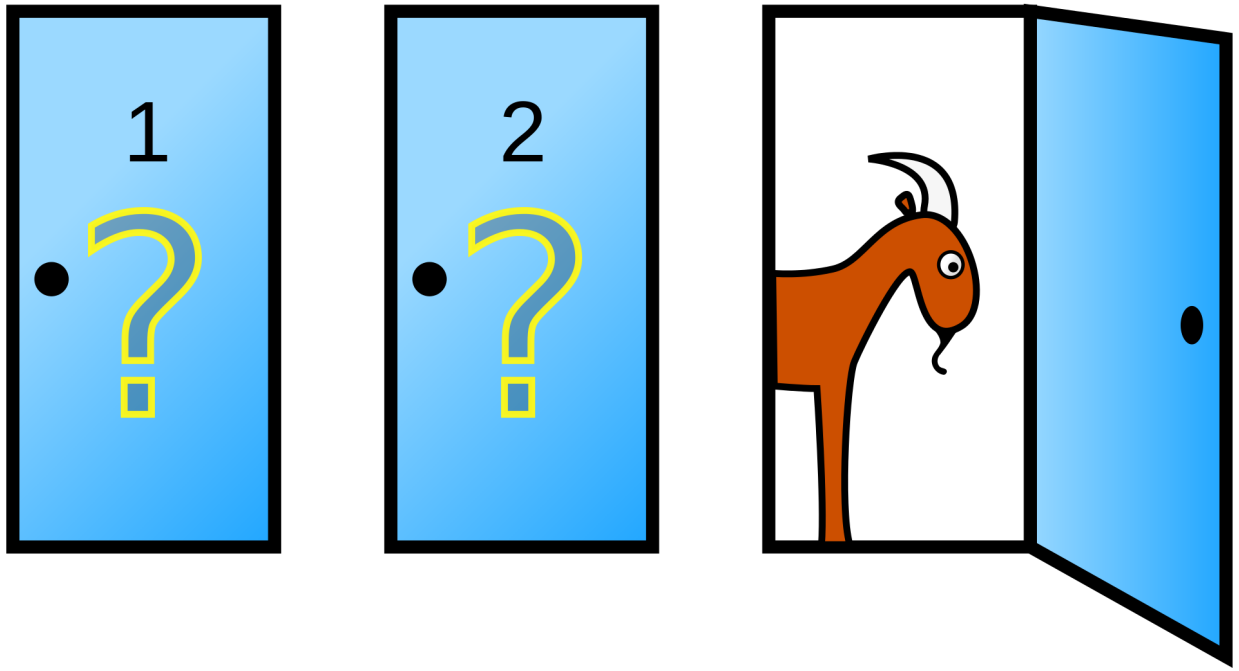
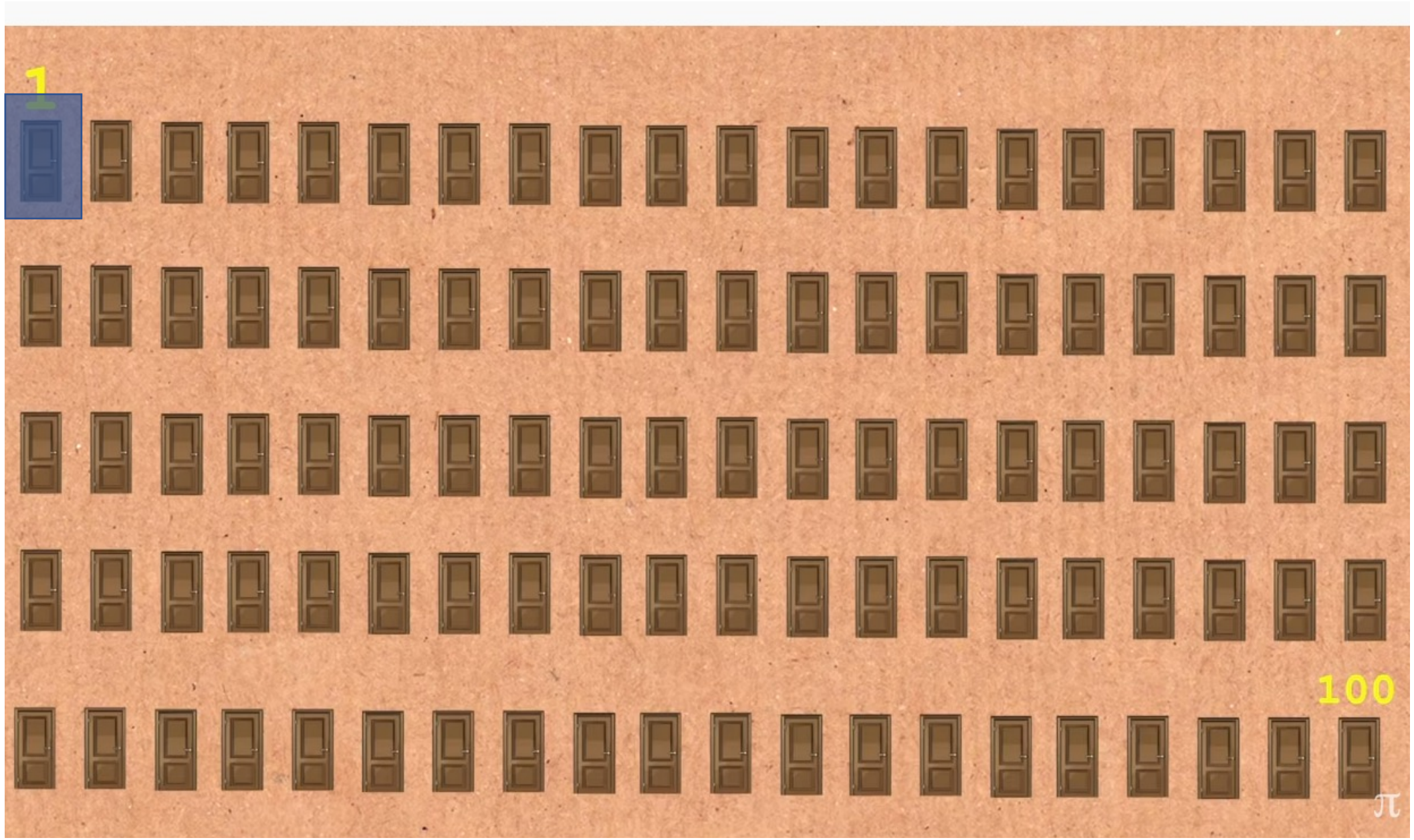
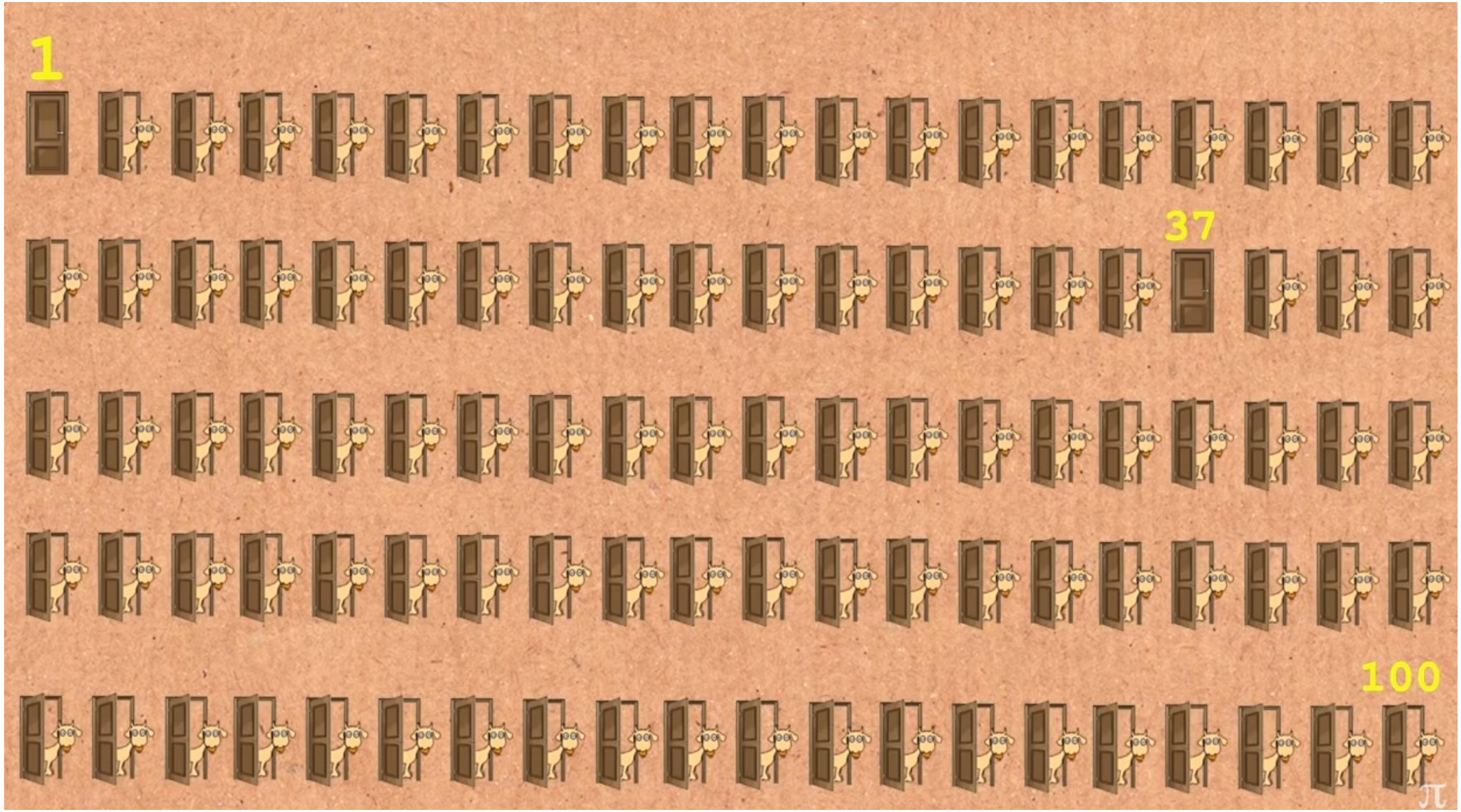
Warn up: **Monty Hall problem**

- Suppose you're on a game show, and you're given the choice of three doors: Behind one door is a car; behind the others, goats. You pick a door, say No. 1, and the host, who knows what's behind the doors, opens another door, say No. 3, which has a goat. He then says to you, "Do you want to pick door No. 2?" Is it to your advantage to switch your choice?
- https://www.youtube.com/watch?v=iBdjqtR2iK4




1
100


1
37
100

Python For Loops

Yijie Wang
yijwang@iu.edu
Computer Science Department
Indiana University Bloomington

Sum of all numbers in a list

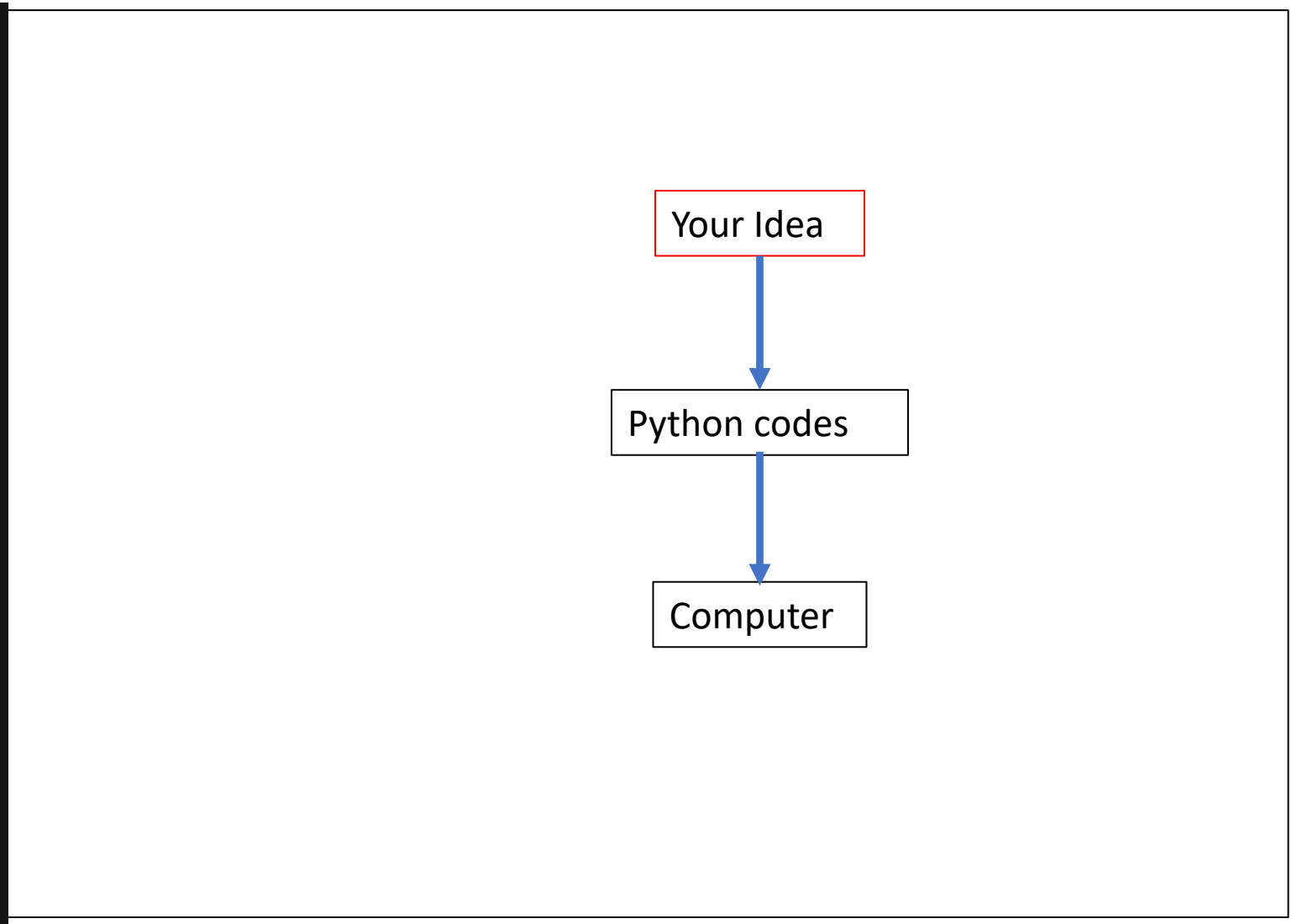
```
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iteration 1
val = numbers[0]
sum = sum + val
# iteration 2
val = numbers[1]
sum = sum + val
# iteration 3
val = numbers[2]
sum = sum + val
# iteration 4
val = numbers[3]
sum = sum + val
# iteration 5
val = numbers[4]
sum = sum + val
# iteration 6
val = numbers[5]
sum = sum + val
# iteration 7
val = numbers[6]
sum = sum + val
# iteration 8
val = numbers[7]
sum = sum + val
# iteration 9
val = numbers[8]
sum = sum + val

print("The sum is", sum)
```



Sum of all numbers in a list

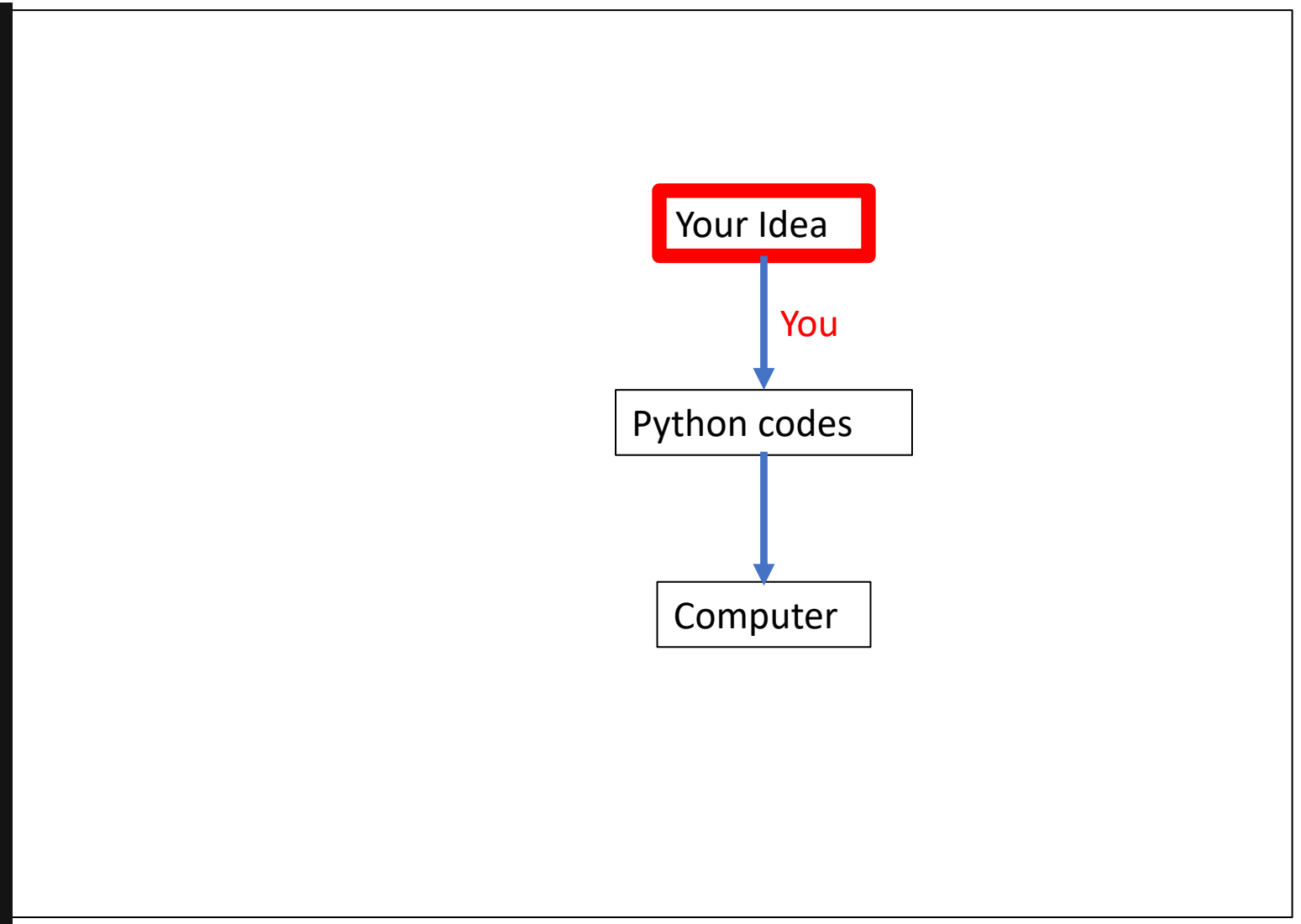
```python
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iteration 1
val = numbers[0]
sum = sum + val
# iteration 2
val = numbers[1]
sum = sum + val
# iteration 3
val = numbers[2]
sum = sum + val
# iteration 4
val = numbers[3]
sum = sum + val
# iteration 5
val = numbers[4]
sum = sum + val
# iteration 6
val = numbers[5]
sum = sum + val
# iteration 7
val = numbers[6]
sum = sum + val
# iteration 8
val = numbers[7]
sum = sum + val
# iteration 9
val = numbers[8]
sum = sum + val

print("The sum is", sum)
```



Sum of all numbers in a list

```
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iteration 1
val = numbers[0]
sum = sum + val
# iteration 2
val = numbers[1]
sum = sum + val
# iteration 3
val = numbers[2]
sum = sum + val
# iteration 4
val = numbers[3]
sum = sum + val
# iteration 5
val = numbers[4]
sum = sum + val
# iteration 6
val = numbers[5]
sum = sum + val
# iteration 7
val = numbers[6]
sum = sum + val
# iteration 8
val = numbers[7]
sum = sum + val
# iteration 9
val = numbers[8]
sum = sum + val

print("The sum is", sum)
```

Sum of all numbers in a list

```
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iteration 1
val = numbers[0]
sum = sum + val
# iteration 2
val = numbers[1]
sum = sum + val
# iteration 3
val = numbers[2]
sum = sum + val
# iteration 4
val = numbers[3]
sum = sum + val
# iteration 5
val = numbers[4]
sum = sum + val
# iteration 6
val = numbers[5]
sum = sum + val
# iteration 7
val = numbers[6]
sum = sum + val
# iteration 8
val = numbers[7]
sum = sum + val
# iteration 9
val = numbers[8]
sum = sum + val

print("The sum is", sum)
```

```
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iterate over the list
for i in range(9):
    val = numbers[i]
    sum = sum+val

print("The sum is", sum)
```

```
Itration 1: sum= 0, val= 6, sum = sum + val =  6
Itration 2: sum= 6, val= 5, sum = sum + val = 11
Itration 3: sum=11, val= 3, sum = sum + val = 14
Itration 4: sum=14, val= 8, sum = sum + val = 22
Itration 5: sum=22, val= 4, sum = sum + val = 26
Itration 6: sum=26, val= 2, sum = sum + val = 28
Itration 7: sum=28, val= 5, sum = sum + val = 33
Itration 8: sum=33, val= 4, sum = sum + val = 37
Itration 9: sum=37, val=11, sum = sum + val = 48
```

```python
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iterate over the list
for i in range(9):
    val = numbers[i]
    sum = sum+val

print("The sum is", sum)
```

```python
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iterate over the list
for val in numbers:
    sum = sum+val

print("The sum is", sum)
```

The range() function

```
print(range(10))

print(list(range(10)))

print(list(range(2, 8)))

print(list(range(2, 20, 3)))
```

```
range(0, 10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[2, 3, 4, 5, 6, 7]
[2, 5, 8, 11, 14, 17]
```

The break Statement

```
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iterate over the list
for val in numbers:
    if sum > 7:
        break
    sum = sum+val

print("The sum is", sum)
```

```
Itration 1: sum= 0, val= 6, sum = sum + val =  6
Itration 2: sum= 6, val= 5, sum = sum + val = 11
```

```
The sum is 11
```

The continue Statement

```
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iterate over the list
for val in numbers:
    if val == 5:
        continue
    sum = sum+val

print("The sum is", sum)
```

```
The sum is 38
```

```
Itration 1: sum= 0, val= 6, sum = sum + val =  6
Itration 2: val==5 --> skip
Itration 3: sum= 6, val= 3, sum = sum + val =  9
Itration 4: sum= 9, val= 8, sum = sum + val = 17
Itration 5: sum=17, val= 4, sum = sum + val = 21
Itration 6: sum=21, val= 2, sum = sum + val = 23
Itration 7: val==5 --> skip
Itration 8: sum=23, val= 4, sum = sum + val = 27
Itration 9: sum=27, val=11, sum = sum + val = 38
```

How to use the for loop to solve scientific problems

- Pierre de Fermat
  - Lawyer, mathematician

How to use the for loop to solve scientific problems

- Pierre de Fermat
  - Lawyer, mathematician

- Conjecture

  For integer n, $2^{2^n} + 1$ is a prime number.

  $n = 1,\ 2^{2^1} + 1 = 5$

  $n = 2,\ 2^{2^2} + 1 = 17$

  $n = 3,\ 2^{2^3} + 1 = 257$

  $n = 4,\ 2^{2^4} + 1 = 65537$

How to use the for loop to solve scientific problems

- Pierre de Fermat
  - Lawyer, mathematician
- Conjecture

  For integer n, $2^{2^n} + 1$ is a prime number.

  $n = 1,\ 2^{2^1} + 1 = 5$

  $n = 2,\ 2^{2^2} + 1 = 17$

  $n = 3,\ 2^{2^3} + 1 = 257$

  $n = 4,\ 2^{2^4} + 1 = 65537$
- Proof wrong by Euler

How to use the for loop to solve scientific problems

- Pierre de Fermat
  - Lawyer, mathematician
- Conjecture

  For integer n, $2^{2^n} + 1$ is a prime number.

  $n = 1,\ 2^{2^1} + 1 = 5$

  $n = 2,\ 2^{2^2} + 1 = 17$

  $n = 3,\ 2^{2^3} + 1 = 257$

  $n = 4,\ 2^{2^4} + 1 = 65537$
- Proof wrong by Euler
  - How? $n = 5,\ 2^{2^5} + 1 = 4294967297$

Fermat's Last Theorem

British mathematician Andrew Wiles.

The blue-eyed islanders puzzle

- On an island, there are **100** people. **95** of them have red eyes, and **5** people have blue eyes.
- It is forbidden for them to know their own eye color, or even to discuss the topic; thus, each resident can (and does) see the eye colors of all other residents, but has no way of discovering his or her own (there are no reflective surfaces).
- If a person does discover his or her own eye color, he or she will kill himself or herself at noon the following day in the village square for all to witness.
- One day, a blue-eyed foreigner visits to the island. Before he leaves, he makes the following public statement: "I am glad to see other blue-eyed persons like myself on the island".
- What is the effect of the statement?

The blue-eyed islanders puzzle

- **Argument 1**. The foreigner has no effect, because his comments do not tell anything that they do not already know (everyone on the island can already see that there are several blue-eyed people in their tribe).
- **Argument 2.** 5 days after the statement, all the blue-eyed people commit suicide.

Check any number n is not a prime number or not

- Definition of the prime number
  - A positive integer greater than 1 which has no other factors except 1 and the number itself is called a prime number.
  - 2, 3, 5, 7 etc. are prime numbers as they do not have any other factors. But 6 is not prime (it is composite) since 2 x 3 = 6.
- Check n is a prime number

Check any number n is not a prime number or not

- Definition of the prime number
  - A positive integer greater than 1 which has no other factors except 1 and the number itself is called a prime number.
  - 2, 3, 5, 7 etc. are prime numbers as they do not have any other factors. But 6 is not prime (it is composite) since 2 x 3 = 6.
- Check n is a prime number
  - Algorithm 1: Check whether 2, 3, 4, 5, 6, …, $n-1$ is a factor of n.

Check any number n is not a prime number or not

- Definition of the prime number
  - A positive integer greater than 1 which has no other factors except 1 and the number itself is called a prime number.
  - 2, 3, 5, 7 etc. are prime numbers as they do not have any other factors. But 6 is not prime (it is composite) since 2 x 3 = 6.
- Check n is a prime number
  - Algorithm 1: Check whether 2, 3, 4, 5, 6, …, $n-1$ is a factor of n.
  - Algorithm 2: Check whether 2, 3, 4, 5, 6, …, $\sqrt{n}$ is a factor of n.
  - Algorithm 3: Check whether n is divisible by 2 or 3, then to check through all numbers of the form $6k \pm 1 \leq \sqrt{n}$.

Algorithm 1

- Check whether 2, 3, 4, 5, 6, …, $n - 1$ is a factor of n.

```
# Program to check if a number is prime or not
num = 4294967297

# define a flag variable
flag = False

# prime numbers are greater than 1
if num > 1:
    # check for factors
    for i in range(2, num):
        if (num % i) == 0:
            # if factor is found, set flag to True
            flag = True
            # break out of loop
            break

# check if flag is True
if flag:
    print(num, "is not a prime number")
else:
    print(num, "is a prime number")
```

The % symbol in Python is called the Modulo Operator.
It returns the remainder of dividing the left hand operand by right hand operand.

Algorithm 2

- Check whether 2, 3, 4, 5, 6, …, $\sqrt{n}$ is a factor of n.

```
# Program to check if a number is prime or not
num = 4294967297

# define a flag variable
flag = False

# prime numbers are greater than 1
if num > 1:
    # check for factors
    for i in range(2, num//2):
        if (num % i) == 0:
            # if factor is found, set flag to True
            flag = True
            # break out of loop
            break

# check if flag is True
if flag:
    print(num, "is not a prime number")
else:
    print(num, "is a prime number")
```

Algorithm 3

- Check whether n is divisible by 2 or 3, then to check through all numbers of the form $6k \pm 1 \leq \sqrt{n}$.

```
# Program to check if a number is prime or not
num = 4294967297

# define a flag variable
flag = False

#Primality test using 6k+-1 optimization.
if num > 1 and num <= 3:
    flag = False
if num % 2 == 0 or num % 3 == 0:
    flag = True

for k in range(6, num//2, 6):
    if num % (k-1) == 0 or num % (k + 1) == 0:
        flag = False
        break
else:
    flag = False

# check if flag is True
if flag:
    print(num, "is not a prime number")
else:
    print(num, "is a prime number")
```

Experiments

- 4294967297
- 16785407
- 1073807359