

Introduction to Programming PYTHON

Chapter 5 – Command: while ... loop

Presenter: **Dr. Nguyen Dinh Long**

Email: dinhlonghcmut@gmail.com

Google-site: <https://sites.google.com/view/long-dinh-nguyen>

Oct. 2022

Programming

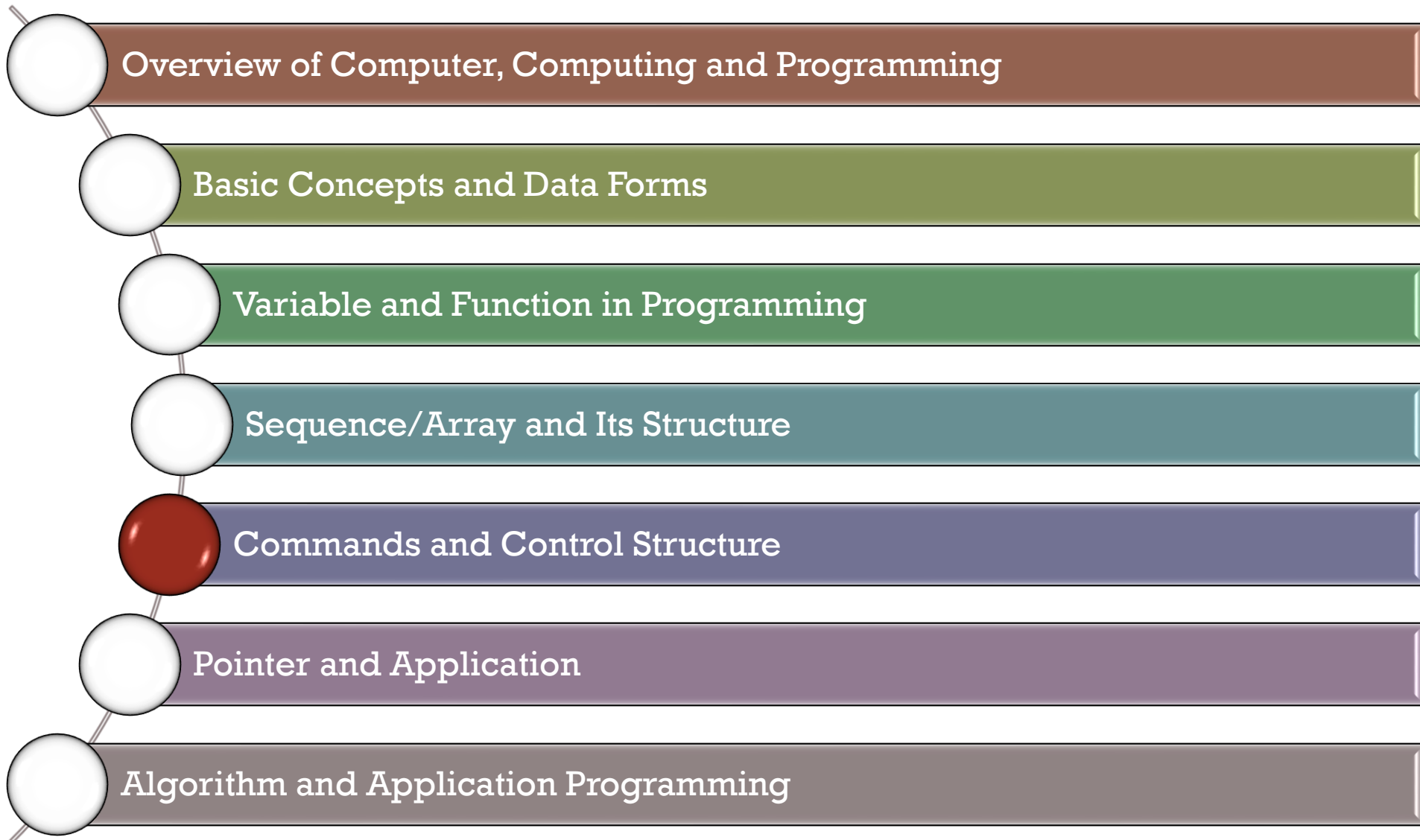
Hamilton was a self-taught programmer, working in the US in the 1960's. Owing to the success of her previous work, Hamilton was the first programmer to be hired for the Apollo project. She became the Director of Software Engineering at the MIT Instrumentation lab. Her lab developed the on-board flight software for NASA's Apollo space project, which took humankind to the moon.

The achievement was a monumental task at a time when computer technology was in its infancy: The astronauts had access to only 72 kilobytes of computer memory (a 256-gigabyte cell phone today carries almost a million times more storage space). Programmers had to use paper punch cards to feed information into room-sized computers with no screen interface.

Margaret Hamilton, NASA's lead software engineer for the Apollo, stands next to the code she wrote by hand that took humanity to the moon in 1969.



Outline



References

Main:

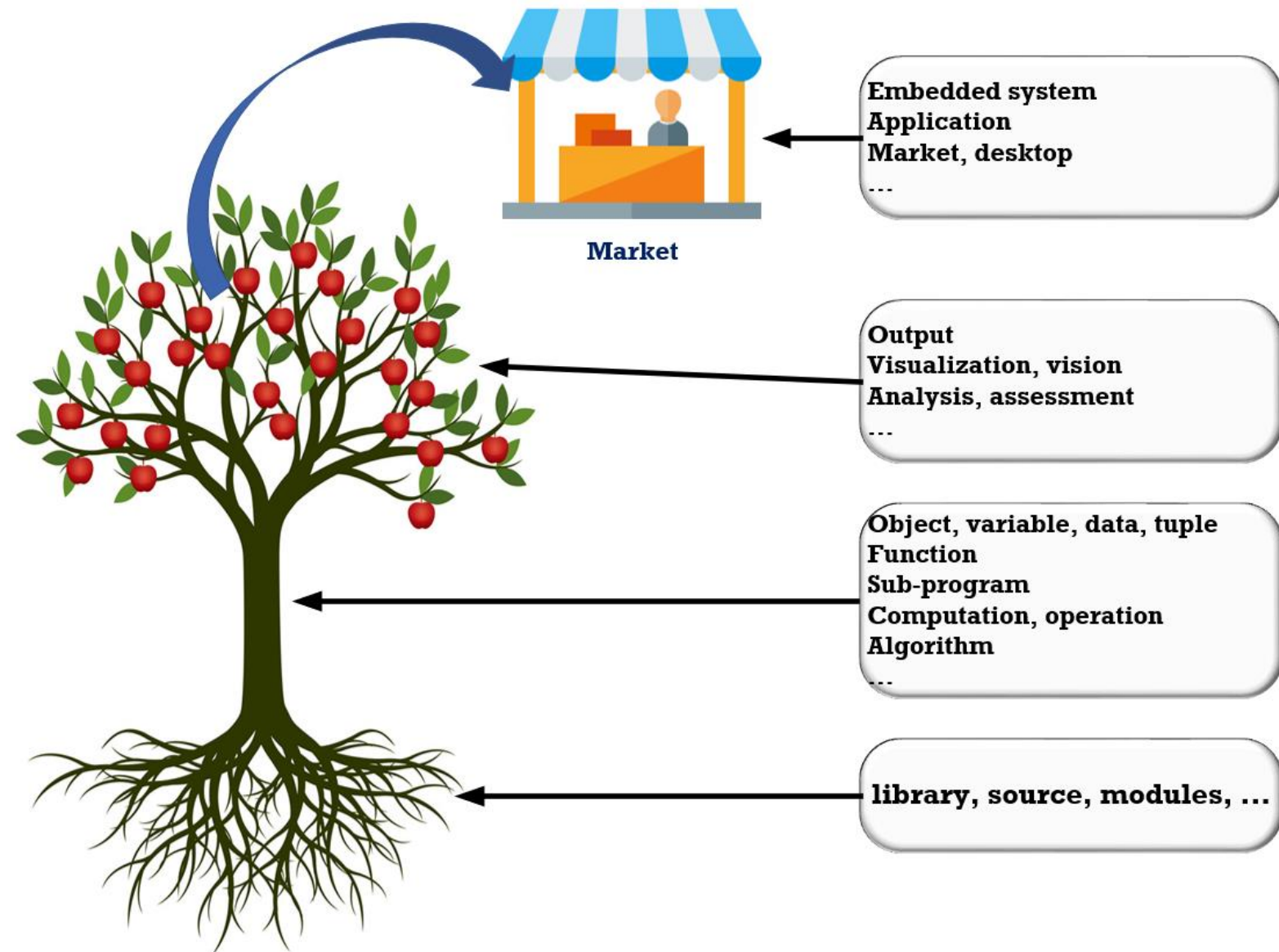
- Maurizio Gabbrielli and Simone Martini, 2010. *Programming Languages: Principles and Paradigms*, Springer.
- Cao Hoàng Trữ, 2004. *Ngôn ngữ lập trình- Các nguyên lý và mô hình*, Nhà xuất bản Đại học Quốc gia Tp. Hồ Chí Minh

More:

- Wes McKinney, 2013. *Python for Data Analysis*, O'Reilly Media.
- Guido van Rossum, Fred L. Drake, Jr., 2012. *The Python Library Reference*, Release 3.2.3.
- Slides here are collected and modified from several sources in Universities and Internet.

Computer programs

General structure:



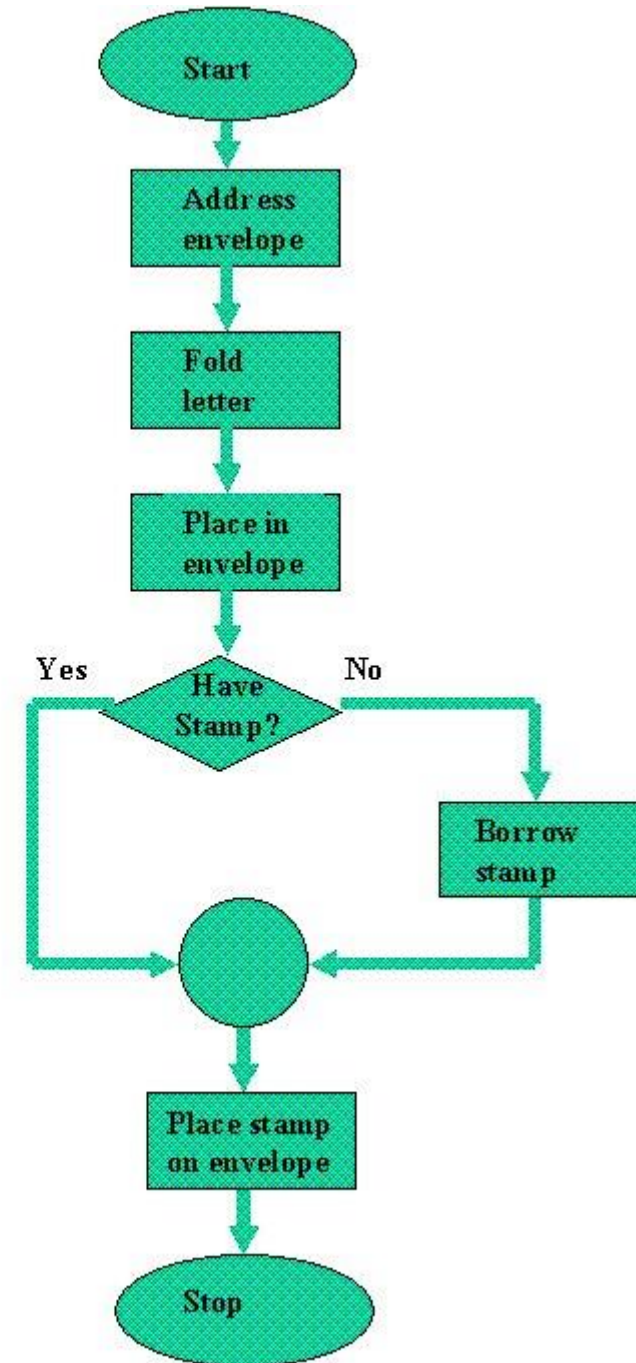
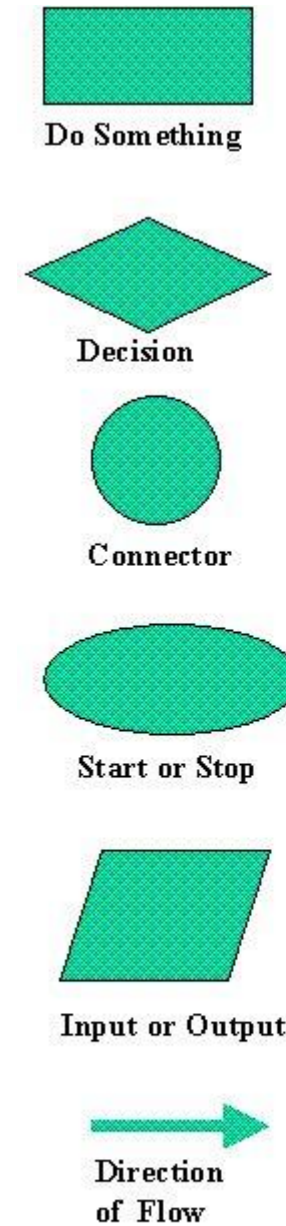
Content of Chapter 5

1. Commands in Python programming
2. Control programs: “for ...” loop and Examples (1 week)
3. Control programs: “while ... do” loop and Examples (1 week)
4. Control programs: “if ... else” loop and Examples (1 week)
5. Examples and Practices: Combine Python loops (1 week)

Structure of Computer programs

Computer programming:

- Objects
- Types
- Variables
- Methods
- Sequences/Arrays



Command – While ... Loop

Python While Loop:

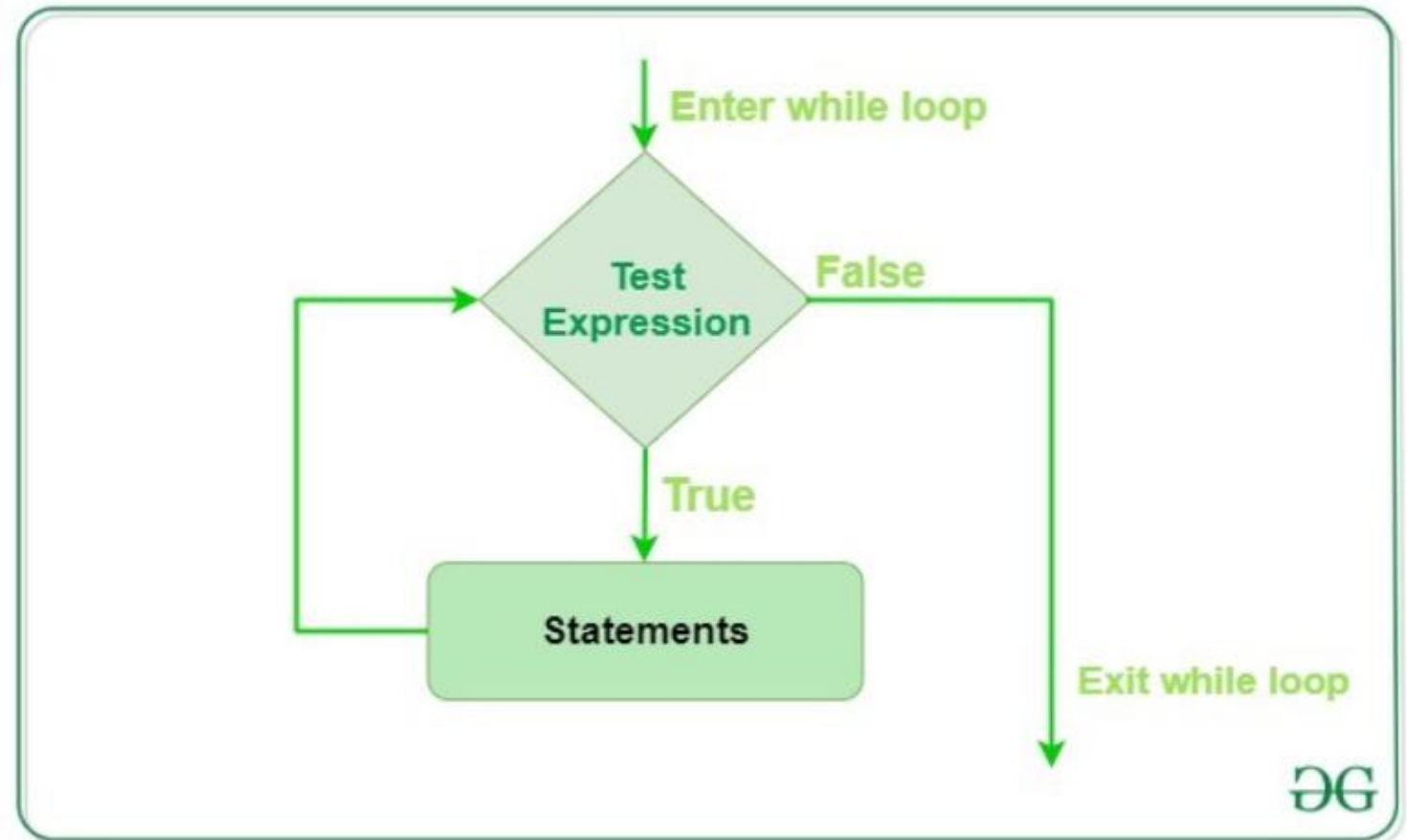
- Python While Loop is used to execute a block of statements repeatedly until a given condition is satisfied. And when the condition becomes false, the line immediately after the loop in the program is executed.

Syntax:

```
while expression:  
    statement(s)
```





While loop falls under the category of indefinite iteration. Indefinite iteration means that the number of times the loop is executed isn't specified explicitly in advance.

Flowchart of While Loop :



Command – While ... Loop

Python While Loop:



```
# Python program to illustrate  
# while loop  
count = 0  
while (count < 3):  
    count = count + 1  
    print("Hello Geek")
```

Output

```
Hello Geek  
Hello Geek  
Hello Geek
```

Command – While ... Loop

❑ Python while loop with list:

- In this example, we will take a list of strings, and iterate over each string using while loop, and print the string length.

```
myList = ['pineapple', 'banana', 'watermelon', 'mango']  
  
index = 0  
while index < len(myList):  
    element = myList[index]  
    print(len(element))  
    index += 1
```

Output

```
9  
6  
10  
5
```

Command – While ... Loop

Python while loop with list:

- We have run a while loop over a list that will run until there is an element present in the list.

```
myList = [5, 7, 8, 3, 4, 2, 9]

index = 0
while index < len(myList):
    element = myList[index]
    if element % 2 == 0:
        print('even')
    else:
        print('odd')
    index += 1
```

Output

```
odd
odd
even
odd
even
even
odd
```

```
# checks if list still
# contains any element
a = [1, 2, 3, 4]

while a:
    print(a.pop())
```

Output

```
4
3
2
1
```

Command – While ... Loop

❑ Python while loop with tuples:

- We defined a tuple with some string values. Then we used while loop to print each item of the tuple..

```
tuple1 = ('a', 'e', 'i', 'o', 'u')

index = 0

while index < len(tuple1):
    #do something
    print(tuple1[index])
    #increment index
    index = index + 1
```

Output

```
a
e
i
o
u
```

Command – While ... Loop

❑ Python while loop with tuples:

- We defined a tuple with some numbers. Then we used while loop to iterate over the Tuple items and sum them.

```
tuple1 = (5, 3, 2, 8, 4, 4, 6, 2)

sum = 0
index = 0


while index < len(tuple1):
    sum = sum + tuple1[index]
    #increment index
    index = index + 1

print(sum)
```


Command – While ... Loop

❑ Single statement while block:

- If there are multiple statements in the block that makes up the loop body, they can be separated by semicolons (;).



```
# Python program to illustrate  
# Single statement while block  
count = 0  
while (count < 5): count += 1; print("Hello Geek")
```

Output:

```
Hello Geek  
Hello Geek  
Hello Geek  
Hello Geek  
Hello Geek
```

Command – While ... Loop

❑ Loop Control Statements:

- Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed. Python supports the following control statements.

- **Continue Statement:**

Python Continue Statement returns the control to the beginning of the loop.

```
# Prints all letters except 'e' and 's'
i = 0
a = 'geeksforgeeks'

while i < len(a):
    if a[i] == 'e' or a[i] == 's':
        i += 1
        continue

    print('Current Letter :', a[i])
    i += 1
```

```
Current Letter : g
Current Letter : k
Current Letter : f
Current Letter : o
Current Letter : r
Current Letter : g
Current Letter : k
```

Command – While ... Loop

❑ Loop Control Statements:

▪ Continue Statement:

Python Continue Statement returns the control to the beginning of the loop.

```
# Python program to show how to use continue loop control

# Initiating the loop
for string in "While Loops":
    if string == "o" or string == "i" or string == "e":
        continue
    print('Current Letter:', string)
```


```
Current Letter: W
Current Letter: h
Current Letter: l
Current Letter:
Current Letter: L
Current Letter: p
Current Letter: s
```

Command – While ... Loop

❏ Loop Control Statements:

▪ Break Statement:

Python Break Statement brings control out of the loop.



```
# break the loop as soon it sees 'e'
# or 's'
i = 0
a = 'geeksforgeeks'

while i < len(a):
    if a[i] == 'e' or a[i] == 's':
        i += 1
        break

    print('Current Letter :', a[i])
    i += 1
```

Output:

Current Letter : g

Command – While ... Loop

❑ Loop Control Statements:

▪ Break Statement:

Python Break Statement brings control out of the loop.

```
# Python program to show how to use the break statement

# Initiating the loop
for string in "Python Loops":
    if string == 'n':
        break
    print('Current Letter: ', string)
```

```
Current Letter: P
Current Letter: y
Current Letter: t
Current Letter: h
Current Letter: o
```


Command – While ... Loop

❏ Loop Control Statements:

▪ Pass Statement:

The Python pass statement to write empty loops. Pass is also used for empty control statements, functions, and classes.

```
# An empty loop
a = 'geeksforgeeks'
i = 0

while i < len(a):
    i += 1
    pass

print('Value of i :', i)
```

Output:

```
Value of i : 13
```

Command – While ... Loop

❑ Loop Control Statements:

▪ Pass Statement:

The Python pass statement to write empty loops. Pass is also used for empty control statements, functions, and classes.

```
# Python program to show how to use the pass statement
for a string in "Python Loops":
    pass
print( 'Last Letter:', string)
```

Output:

```
Last Letter: s
```

Command – While ... Loop

❑ Loop Control Statements:

▪ While loop with else:

As discussed above, while loop executes the block until a condition is satisfied. When the condition becomes false, the statement immediately after the loop is executed. The else clause is only executed when your while condition becomes false. If you break out of the loop, or if an exception is raised, it won't be executed.

```
# Python program to demonstrate
# while-else loop

i = 0
while i < 4:
    i += 1
    print(i)
else: # Executed because no break in for
    print("No Break\n")

i = 0
while i < 4:
    i += 1
    print(i)
    break
else: # Not executed as there is a break
    print("No Break")
```

Output:

```
1
2
3
4
No Break



1
```

Command – While ... Loop

❑ Loop Control Statements:

▪ Sentinel Controlled Statement:

We don't use any counter variable because we don't know that how many times the loop will execute. Here user decides that how many times he wants to execute the loop. For this, we use a sentinel value. A sentinel value is a value that is used to terminate a loop whenever a user enters it, generally, the sentinel value is -1..

```
 a = int(input('Enter a number (-1 to quit): '))  
 while a != -1:  
    a = int(input('Enter a number (-1 to quit): '))
```

Explanation:

- First, it asks the user to input a number. if the user enters -1 then the loop will not execute
- User enter 6 and the body of the loop executes and again ask for input
- Here user can input many times until he enters -1 to stop the loop
- User can decide how many times he wants to enter input

Command – While ... Loop

❑ Loop Control Statements:

▪ Nested While Loop:

If a while loop consists of another while loop we can call it a nested while loop. There is no particular limit for number of loops in a nested while loop. It may go on for as many times a user requires or declares it in the program.

```
i = 1
j = 5
while i < 6:
    while j > 0:
        print(i,j)
        j = j - 1
        i = i + 1
```

Output:

```
1 5
2 4
3 3
4 2
5 1
```

```
i = 'edureka'
j = 1
while j > 0:
    for x in i:
        print(j , x)
        j = j + 1
    if x == 'a':
        break
```

Output:

```
1 e
2 d
3 u
4 r
5 e
6 k
7 a
```


Command – While ... Loop

❑ Loop Control Statements:

▪ Infinite While ... Loop:

A loop becomes infinite loop if a condition never becomes FALSE. You must use caution when using while loops because of the possibility that this condition never resolves to a FALSE value. This results in a loop that never ends. Such a loop is called an infinite loop.

An infinite loop might be useful in client/server programming where the server needs to run continuously so that client programs can communicate with it as and when required.

```
var = 1
while var == 1 : # This constructs an infinite loop
    num = raw_input("Enter a number :")
    print "You entered: ", num

print "Good bye!"
```

```
Enter a number :20
You entered: 20
Enter a number :29
You entered: 29
Enter a number :3
You entered: 3
Enter a number between :Traceback (most recent call last):
  File "test.py", line 5, in <module>
    num = raw_input("Enter a number :")
KeyboardInterrupt
```

Python While ... Loop

Examples

Command – While ... Loop

❏ Python While Loop - Examples:

- **Sum of squares** of the first 15 natural numbers using a while loop.

```
# Python program example to show the use of while loop

num = 15

# initializing summation and a counter for iteration
summation = 0
c = 1

while c <= num: # specifying the condition of the loop
    # beginning the code block
    summation = c**2 + summation
    c = c + 1 # incrementing the counter

# print the final sum
print("The sum of squares is", summation)
```

Output:

```
The sum of squares is 1240
```

Command – While ... Loop

Python While Loop - Examples:

- We will construct a Python program to verify if the given integer is a **prime number** or not.

```
num = [34, 12, 54, 23, 75, 34, 11]

def prime_number(number):
    condition = 0
    iteration = 2
    while iteration <= number / 2:
        if number % iteration == 0:
            condition = 1
            break
        iteration = iteration + 1

    if condition == 0:
        print(f"{number} is a PRIME number")
    else:
        print(f"{number} is not a PRIME number")

for i in num:
    prime_number(i)
```

Output:

```
34 is not a PRIME number
12 is not a PRIME number
54 is not a PRIME number
23 is a PRIME number
75 is not a PRIME number
34 is not a PRIME number
11 is a PRIME number
```

Command – While ... Loop

❏ Python While Loop - Examples:

- **Multiplication Table** using While Loop. We will use the while loop for printing the multiplication table of a given number.

```
num = 21
counter = 1
# we will use a while loop for iterating 10 times for the multiplication table
print("The Multiplication Table of: ", num)
while counter <= 10: # specifying the condition
    ans = num * counter
    print (num, 'x', counter, '=', ans)
    counter += 1 # expression to increment the counter
```

The Multiplication Table of: 21

21 x 1 = 21

21 x 2 = 42

21 x 3 = 63

21 x 4 = 84

21 x 5 = 105

21 x 6 = 126

21 x 7 = 147

21 x 8 = 168

21 x 9 = 189

21 x 10 = 210

Command – While ... Loop

❏ Python While Loop - Examples:

- We will use a Python while loop to square every number of **a list**.

```
# Python program to square every number of a list
# initializing a list
list_ = [3, 5, 1, 4, 6]
squares = []
# programing a while loop
while list_: # until list is not empty this expression will give boolean True after that False
    squares.append( (list_.pop())**2)
# print the squares
print( squares )
```

```
[36, 16, 1, 25, 9]
```

Command – While ... Loop

Python While Loop - Examples:

- We'll need to **recruit logical operators** to combine two or more expressions specifying conditions into a single while loop. This instructs Python on collectively analyzing all of the given expressions of conditions.
- We can construct a while loop with multiple conditions in this example. We have given two conditions and a **and** keyword, meaning until both conditions give boolean True, the loop will execute the statements.

```
num1 = 17
num2 = -12

while num1 > 5 and num2 < -5 : # multiple conditions in a single while loop
    num1 -= 2
    num2 += 3
    print( (num1, num2) )
```

Output:

```
(15, -9)
(13, -6)
(11, -3)
```

Command – While ... Loop

❏ Python While Loop - Examples:

- We'll need to **recruit logical operators** to combine two or more expressions specifying conditions into a single while loop. This instructs Python on collectively analyzing all of the given expressions of conditions.
- We can construct a while loop with multiple conditions in this example. We have given two conditions and a **and** keyword, meaning until both conditions give boolean True, the loop will execute the statements.

```
num1 = 17
num2 = -12

while num1 > 5 and num2 < -5 :
    num1 -= 2
    num2 += 3
    print( (num1, num2) )
```

Output:

```
(15, -9)
(13, -6)
(11, -3)
(9, 0)
(7, 3)
(5, 6)
```

Command – While ... Loop

❑ Python While Loop - Examples:

- We'll need to **recruit logical operators** to combine two or more expressions specifying conditions into a single while loop. This instructs Python on collectively analyzing all of the given expressions of conditions.
- We can construct a while loop with multiple conditions in this example. We have given two conditions and a **and** keyword, meaning until both conditions give boolean True, the loop will execute the statements.

```
num1 = 9
num = 14
maximum_value = 4
counter = 0
while (counter < num1 or counter < num2) and not counter >= maximum_value: # grouping multiple conditions
    print(f"Number of iterations: {counter}")
    counter += 1
```

Output:

```
Number of iterations: 0
Number of iterations: 1
Number of iterations: 2
Number of iterations: 3
```

Command – While ... Loop

❏ Python While Loop – Examples (Arrays):

- The [for loop](#), is frequently used for iterating elements of an array, [while loop](#) on the other hand, is capable of serving several purposes, and it can iterate an array too.

```
myList = [8, 9, 2, 3, 4, 6]
i = 0
while i < len(myList):
    print (myList[i])
    i = i+1;
```

```
bigList = [[1, 3, 6], [8, 2,], [0, 4, 7, 10], [1, 5, 2], [6]]
i = 0
j = 0
while i<len(bigList):
    while j<len(bigList[i]):
        print ("Element of list within a list -",bigList[i][j])
        j=j+1;
    i=i+1
```

Command – While ... Loop

Python While Loop – Examples ([Arrays](#)):

- Using a loop, calculate the average of the first 10 positive, non-zero integers.

While loop:

```
count = 1
result = 0
while count <= 10:
    result += count
    count += 1

print(result / (count - 1))
```

5.5

Command – While ... Loop

❑ Python While Loop – Examples (Arrays):

- Given two lists of the same length: ListA = [2, 4, 6, 8, 10] and ListB = [25, 30, 35, 40, 45].
- Create a new list ListC to store the products of the corresponding elements of ListA and ListB.

While loop:

```
ListC = []
ListA = [2,4,6,8,10]
ListB = [25,30,35,40,45]
count = 0
while count < len(ListA):
    ListC.append(ListA[count] * ListB[count])
    count += 1

print(ListC)
```

[50, 120, 210, 320, 450]

Command – While ... Loop

Python While Loop – Examples (Arrays):

- Using a loop, calculate the value of the factorial of 5 → $5! = 5 \times 4 \times 3 \times 2 \times 1$.

While loop:

```
result = 1
count = 1
while count <= 5:
    result *= count
    count += 1
print(result)
```

120

Command – While ... Loop and Comparison

Python While Loop and For Loop:

- Choosing Your Looping Construct
 - List comprehension – Generating a new list.
 - for loop – Performing some action across each element of an existing list.
 - while loop – Performing some action repeatedly until a condition is met without knowing the explicit boundaries upfront.

While loop:

```
NumberList = []
count = 1
while count <= 20:
    if count % 2 == 0:
        NumberList.append(count)
    count += 1
print(NumberList)
```

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]

For loop:

```
NumberList = []
for count in range(1,21):
    if count % 2 == 0:
        NumberList.append(count)
print(NumberList)
```

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]

List Comprehension:

```
NumberList = [count for count in range(1,21) if count % 2 == 0]
print(NumberList)
```

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]

Command – While ... Loop

❑ Python While Loop – Examples ([Arrays](#)):

- Using a loop, calculate the value of the power of two → $[2^n = 2 \times 2 \times \dots \times 2, n \text{ times}]$.

Python While ... Loop

A story ...

Wheat and Chessboard problem

Câu chuyện ô bàn cờ và số hạt thóc vĩ đại

Chuyện kể rằng, vào thế kỉ thứ 6, ở vương quốc Gupta, (thuộc Ấn Độ) đức vua Sêram trị vì đất nước. Khi đất nước thái bình, muốn tìm kiếm một thú vui tao nhã nào đó để tận hưởng cuộc sống nên đức vua ban bố khắp thiên hạ ai có sáng kiến gì hay thì dâng lên nhà vua. Một hôm, có một nhà thông thái tên là Seta đã dâng lên nhà vua một trò chơi có tên là Saturanga.

Lúc đầu, nhà vua nhìn thấy bàn cờ 64 ô với những quân cờ và nước đi đơn giản nên tỏ ra không hứng thú lắm. Nhưng khi nhà thông thái hướng dẫn cách chơi và cùng chơi với nhà vua đã khiến cho nhà vua vô cùng khâm phục sự sắc sảo và đa dạng của các tình huống trên bàn cờ. Nhà vua đã say mê chơi trò ấy suốt cả tuần trời mà không thấy chán. Quá khâm phục trí tuệ của người sáng tạo, nhà vua quyết định ban thưởng hậu hĩnh. Ngài nói với Seta:

– Ta muốn tặng thưởng cho ông một cách xứng đáng. Nào, Seta, ông hãy nói điều ông muốn ta ban thưởng cho ông. Đừng ngại, hãy nói ra phần thưởng có thể làm ông thỏa mãn và ông sẽ nhận được nó ngay lập tức. Ta rất giàu có và không tiếc gì để có thể ban thưởng cho ông một cách xứng đáng.

Seta im lặng suy nghĩ. Lúc sau ông chậm rãi thưa:

– Thưa đức vua, lòng tốt của ngài thật vĩ đại. Nhưng vì đột ngột quá nên thần chưa nghĩ ra được điều gì. Xin ngài cho thần nghĩ đến sáng ngày mai.

Sáng sớm hôm sau, như đã hứa, Seta vào gặp nhà vua. Seta cung kính thưa:

– Thưa đức vua! Thần không ham gì vàng bạc châu báu. Chỉ xin ngài thực hiện cho thần một điều ước.

– Ngài hãy ra lệnh cho đặt 1 hạt lúa mì vào ô thứ nhất của bàn cờ.

– Tiếp đến, ở ô thứ 2, ngài hãy đặt vào 2 hạt thóc.

– Cứ như cách làm vừa rồi, ô tiếp sau sẽ có số thóc gấp đôi ô trước cho đến ô cuối cùng. Thần sẽ nhận lấy toàn bộ số thóc đó.

Seta cảm ơn và xin phép ra về, hẹn đến ngày nhà vua đếm xong số thóc sẽ đến nhận lấy.









Câu chuyện ô bàn cờ và số hạt thóc vĩ đại



Câu chuyện ô bàn cờ và số hạt thóc vĩ đại

$2^1 = 2$
 $2^2 = 4$
 $2^3 = 8$
 $2^4 = 16$
 $2^5 = 32$
 $2^6 = 64$
 $2^7 = 128$
 $2^8 = 256$

$2^9 = 512$
 $2^{10} = 1024$
 $2^{11} = 2048$
 $2^{12} = 4096$
 $2^{13} = 8192$
 $2^{14} = 16384$
 $2^{15} = 32768$
 $2^{16} = 65536$

							
1	2	4	8	16	32	64	128
256	512	1024	2048	4096	8192	16384	32768
256	512	1,024	2,048	4,096	8,192	16,384	32,768
65536	131K	262K	524K	1.05M	2.10M	4.19M	8.39M
65,536	131,072	262,144	524,288	1,048,576	2,097,152	4,194,304	8,388,608
16.8M	33.6M	67.1M	134M	268M	537M	1.07G	2.15G
16,777,216	33,554,432	67,108,864	134,217,728	268,435,456	536,870,912	1,073,741,824	2,147,483,648
4.29G	8.59G	17.2G	34.4G	68.7G	137G	275G	550G
4,294,967,296	8,589,934,592	17,179,869,184	34,359,738,368	68,719,476,736	137,438,953,472	274,877,906,944	549,755,813,888
1.10T	2.20T	4.40T	8.80T	17.6T	35.2T	70.4T	141T
1,099,511,627,776	2,199,023,255,552	4,398,046,511,104	8,796,093,022,208	17,592,186,044,416	35,184,372,088,832	70,368,744,177,664	140,737,488,355,328
281T	563T	1.13P	2.25P	4.50P	9.01P	18.0P	36.0P
281,474,976,710,656	562,949,953,421,312	1,125,899,906,842,624	2,251,799,813,685,248	4,503,599,627,370,496	9,007,199,254,740,992	18,014,398,509,481,984	36,028,797,018,963,968
72.1P	144P	288P	576P	1.15E	2.31E	4.61E	9.22E
72,057,594,037,927,936	144,115,188,075,855,872	288,230,376,151,711,744	576,460,752,303,423,488	1,152,921,504,606,846,976	2,305,843,009,213,693,952	4,611,686,018,427,387,904	9,223,372,036,854,775,808

Câu chuyện ô bàn cờ và số hạt thóc vĩ đại

On the entire chessboard there would be $2^{64} - 1 = 18,446,744,073,709,551,615$ grains of wheat, weighing about 1,199,000,000,000 metric tons. This is about 1,645 times the global production of wheat (729,000,000 metric tons in 2014 and 780.8 million tonnes in 2019).

A statistic depicts the average annual prices for U.S. wheat (HRW). In 2020, the average price for U.S. wheat (HRW) stood at **211 U.S. dollars** per metric ton.

The price for 1,199,000,000,000 tons = \$253 billion USD

6.2 billion billion vnd (1 USD = 24,475 vnd)

Bài tập thực hành



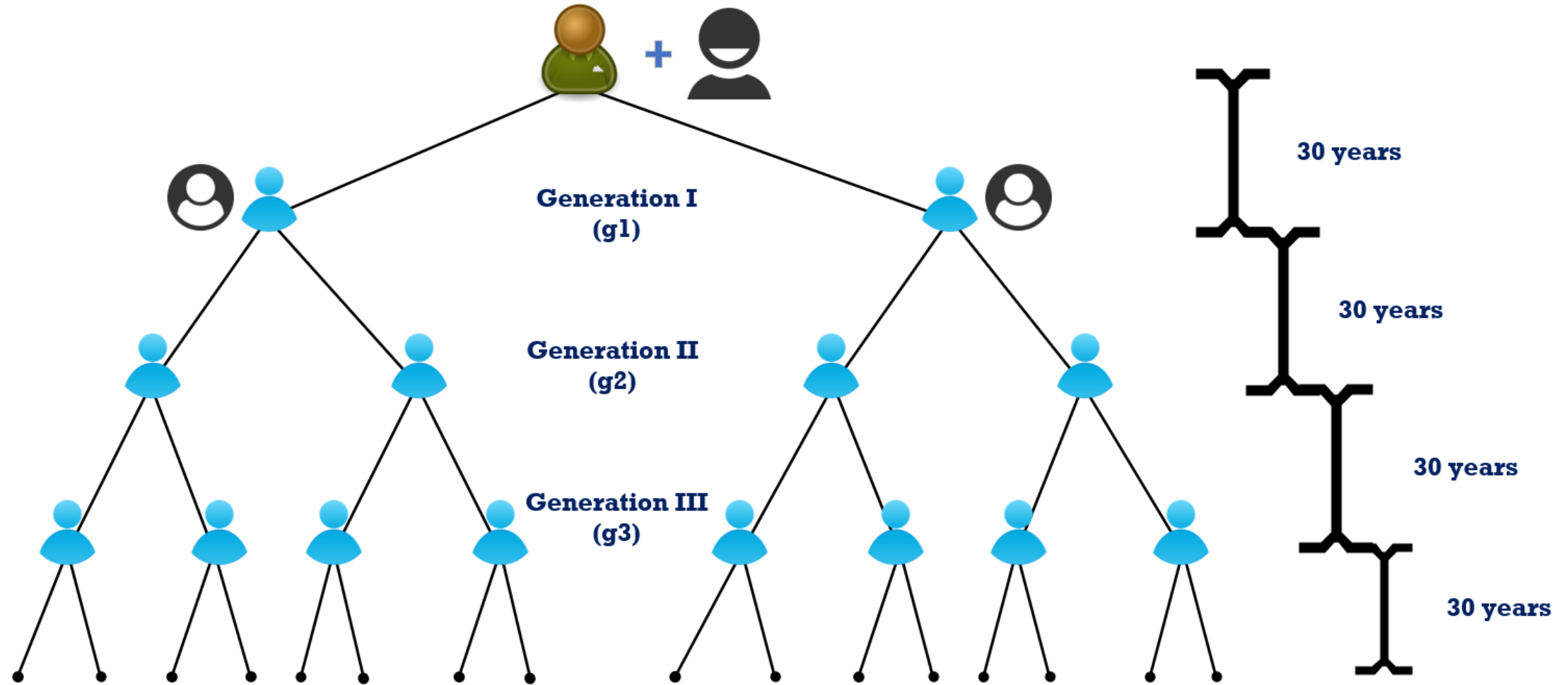
$$2^0 = 1$$

$$2^1 = 2$$

$$2^2 = 4$$

$$2^3 = 8$$

$$2^4 = 16$$



Bài tập thực hành



Viết một chương trình tính toán và hiển thị số lượng thành viên liên quan huyết thống của bạn (con cháu) tại thế hệ cuối cùng (g_n) theo số năm khảo sát bất kỳ.

- Tạo input function nhập số năm bất kỳ (“so_nam”), tính toán sẽ tạo ra bao nhiêu thế hệ theo mô hình khảo sát từ so-nam đã có.
- Viết function tính toán hàm power of two (2^n), tự động tính toán và hiển thị kết quả giá trị theo ngõ vào (giá trị “so_nam” ở input function).
- Plot số lượng thành viên gia phả theo từng thế hệ cho đến khi kết thúc giá trị “so_nam” từ input function.

Start/Stop

Input/Output

Do something

Decision

+

-

