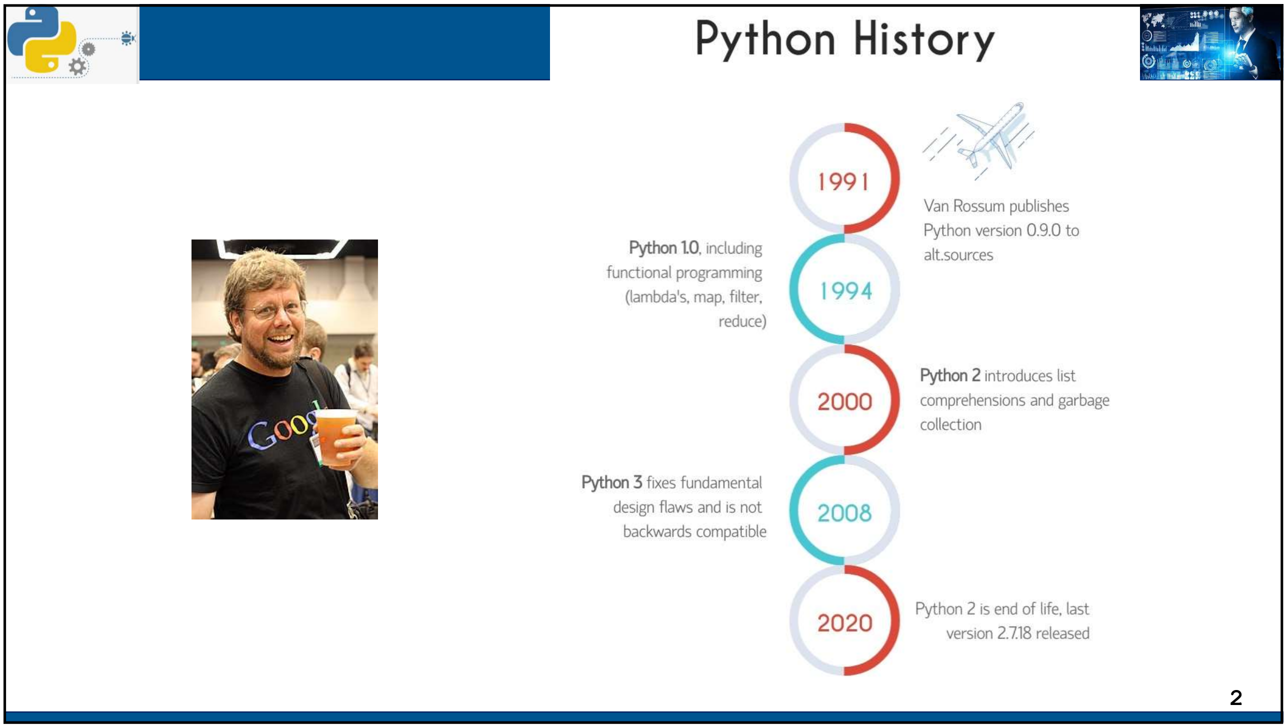
Introduction to Python Programing Language

1



Python History

1991 — Van Rossum publishes Python version 0.9.0 to alt.sources

1994 — **Python 1.0**, including functional programming (lambda's, map, filter, reduce)

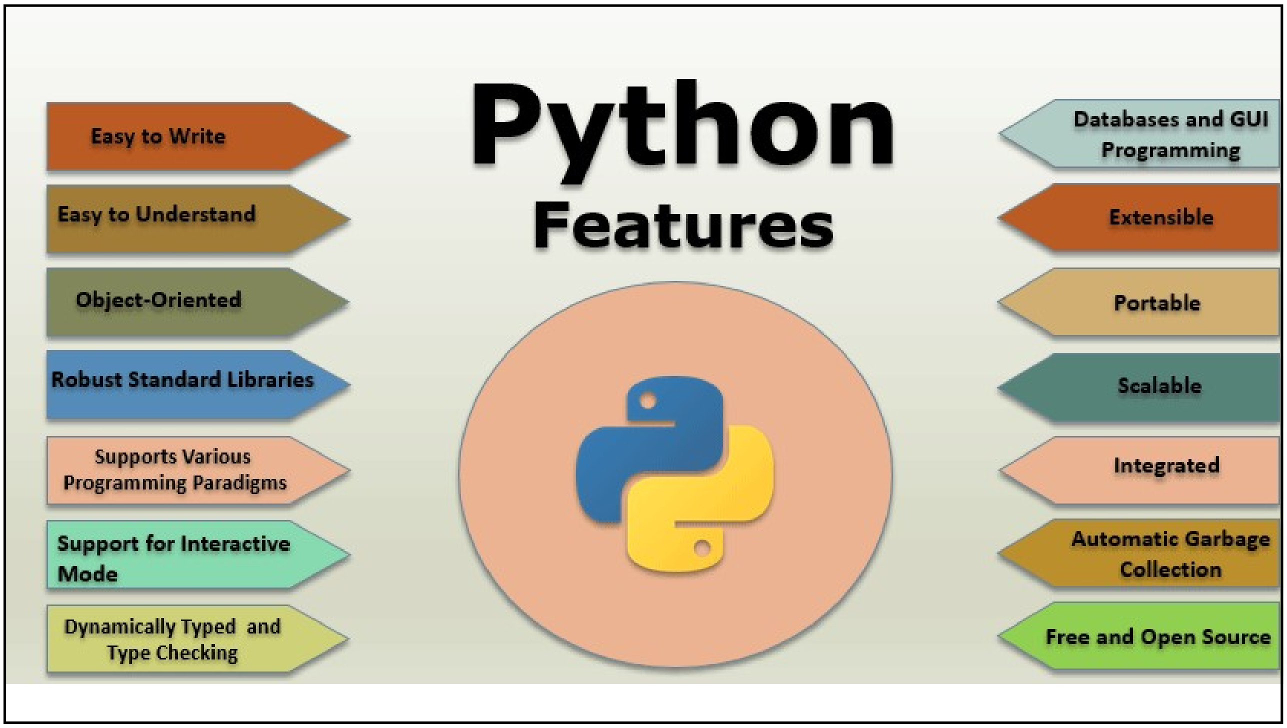
2000 — **Python 2** introduces list comprehensions and garbage collection

2008 — **Python 3** fixes fundamental design flaws and is not backwards compatible

2020 — Python 2 is end of life, last version 2.7.18 released

2

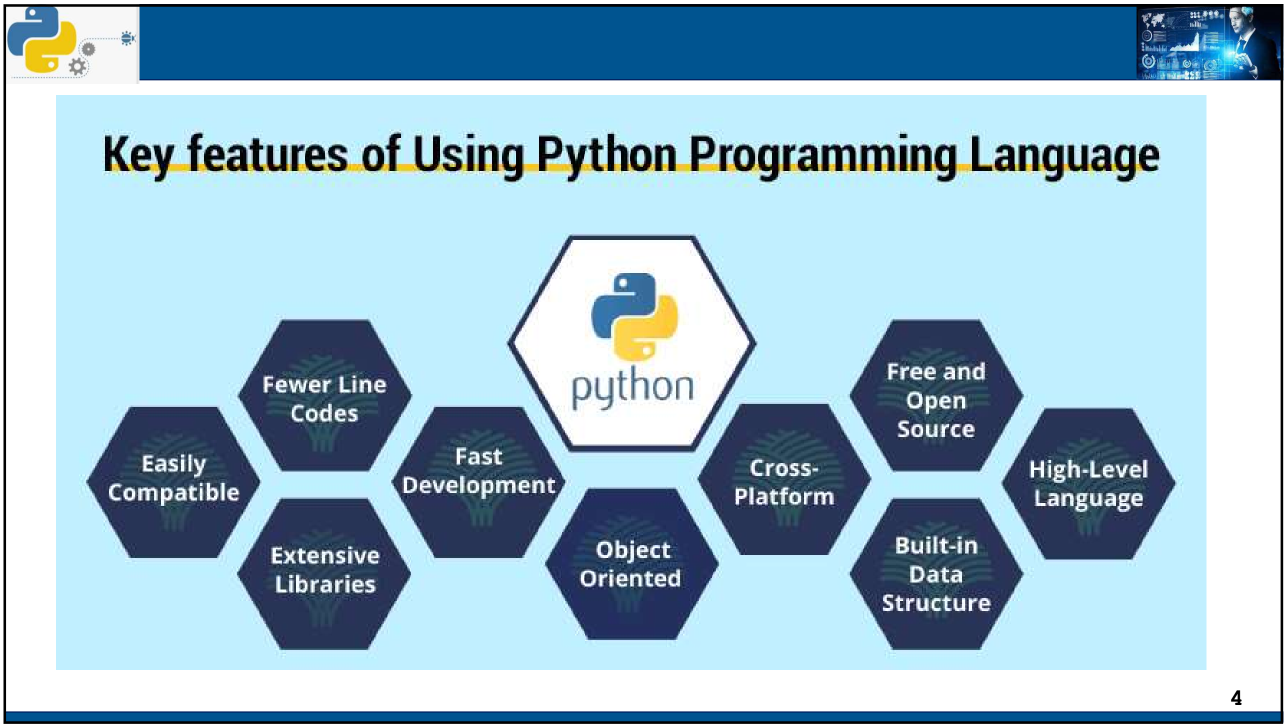
2



Python Features

- Easy to Write
- Easy to Understand
- Object-Oriented
- Robust Standard Libraries
- Supports Various Programming Paradigms
- Support for Interactive Mode
- Dynamically Typed and Type Checking
- Databases and GUI Programming
- Extensible
- Portable
- Scalable
- Integrated
- Automatic Garbage Collection
- Free and Open Source

3

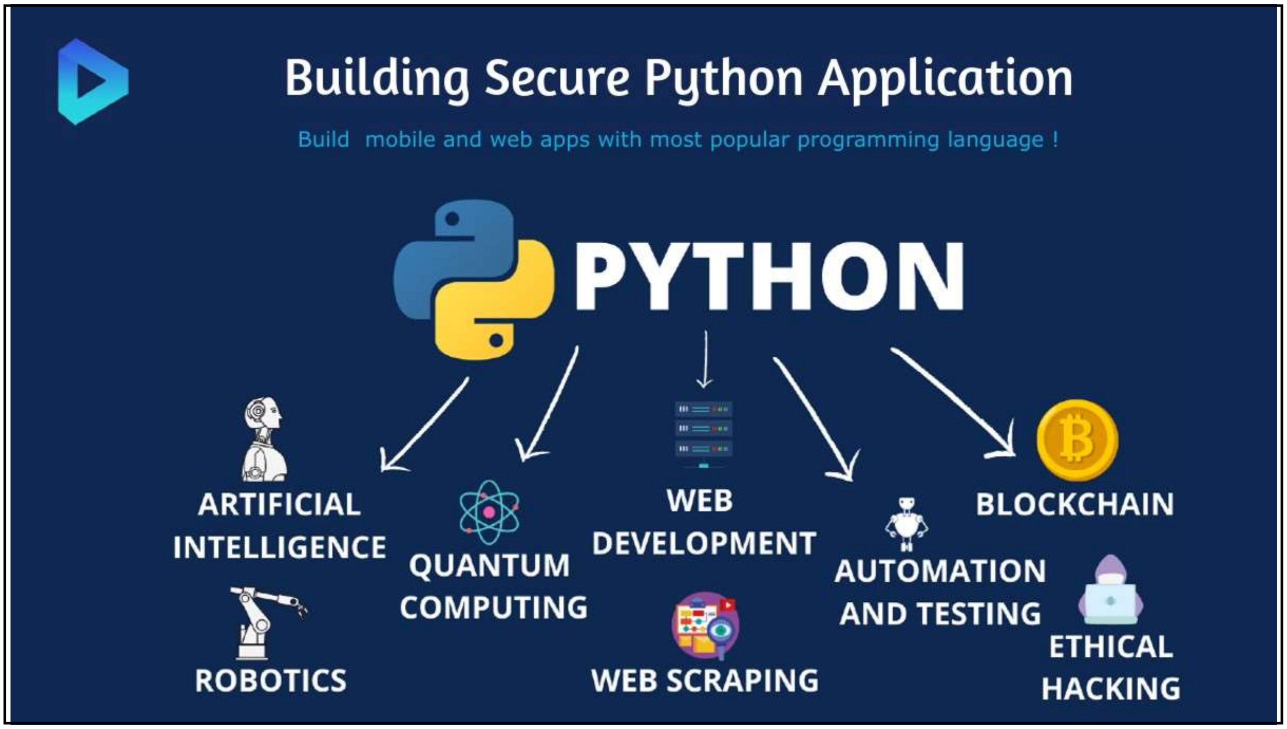


Key features of Using Python Programming Language

- Easily Compatible
- Fewer Line Codes
- Extensive Libraries
- Fast Development
- Object Oriented
- Cross-Platform
- Free and Open Source
- Built-in Data Structure
- High-Level Language

4

4



Building Secure Python Application

Build mobile and web apps with most popular programming language !

PYTHON

ARTIFICIAL INTELLIGENCE

QUANTUM COMPUTING

WEB DEVELOPMENT

AUTOMATION AND TESTING

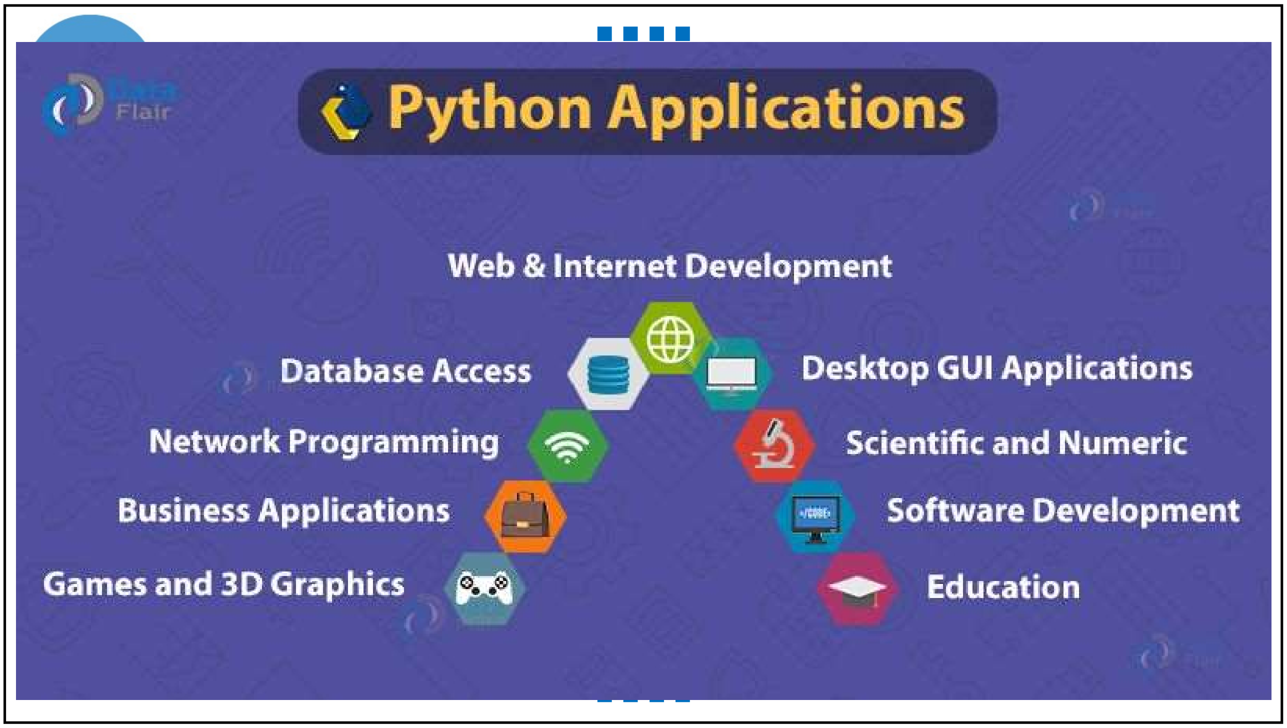
BLOCKCHAIN

ROBOTICS

WEB SCRAPING

ETHICAL HACKING

5



Python Applications

Web & Internet Development

Database Access

Desktop GUI Applications

Network Programming

Scientific and Numeric

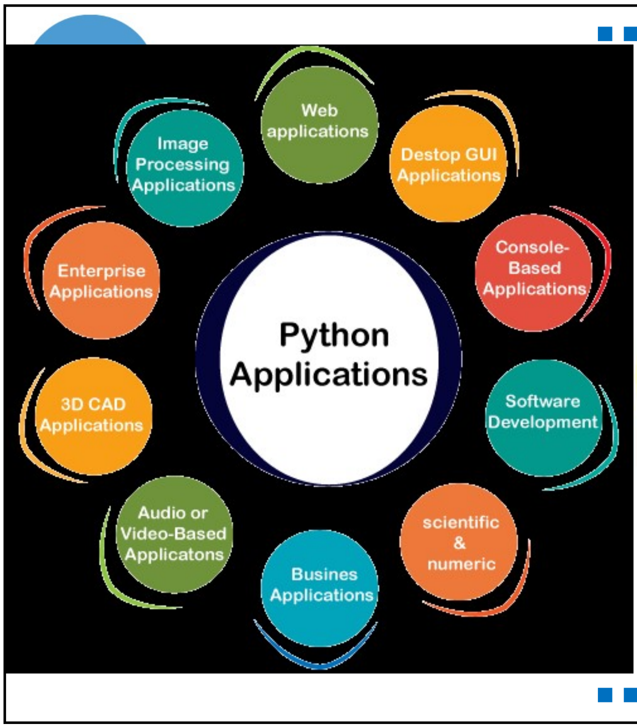
Business Applications

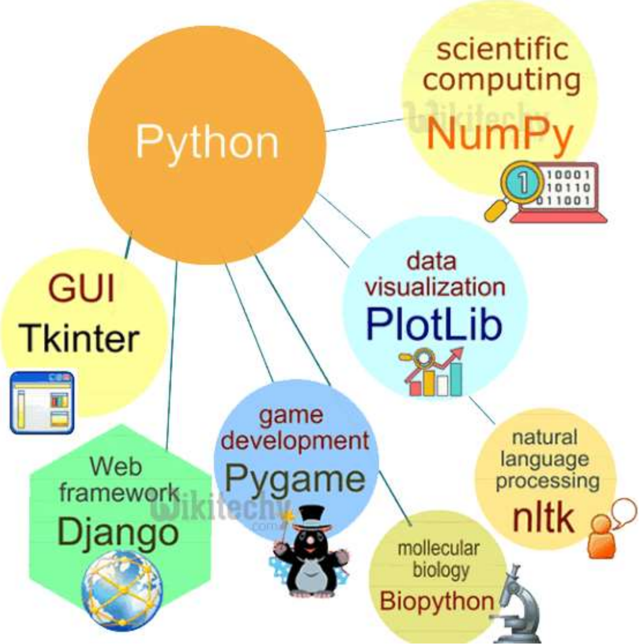
Software Development

Games and 3D Graphics

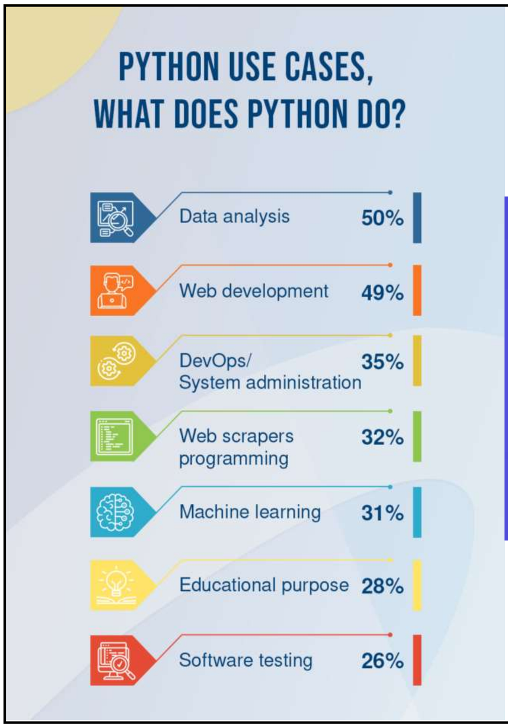
Education

6



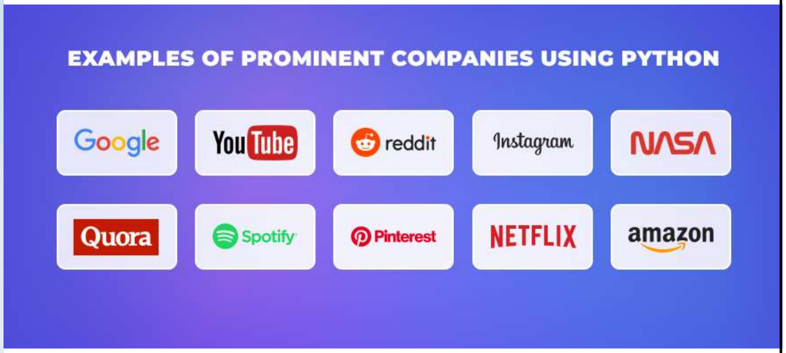


7



PYTHON USE CASES, WHAT DOES PYTHON DO?

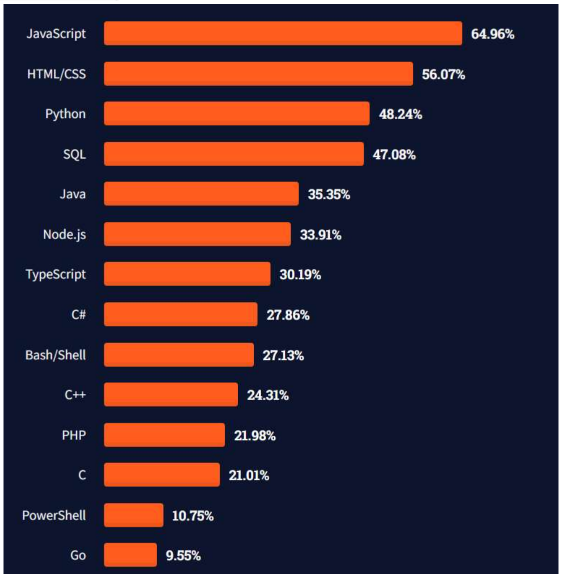
Data analysis	50%
Web development	49%
DevOps/ System administration	35%
Web scrapers programming	32%
Machine learning	31%
Educational purpose	28%
Software testing	26%

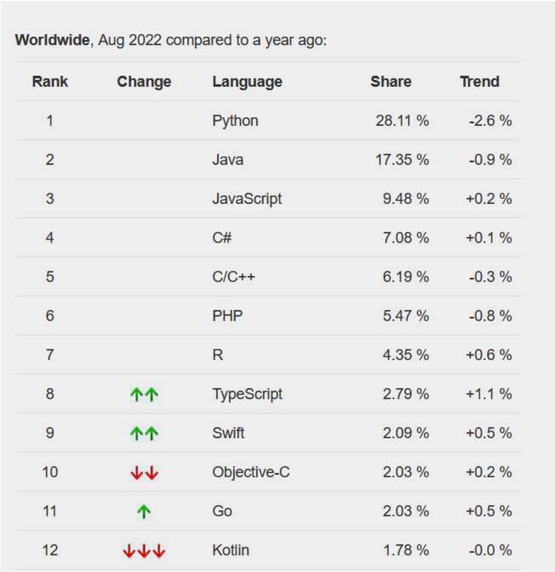


EXAMPLES OF PROMINENT COMPANIES USING PYTHON

8

Most popular language programming 2021





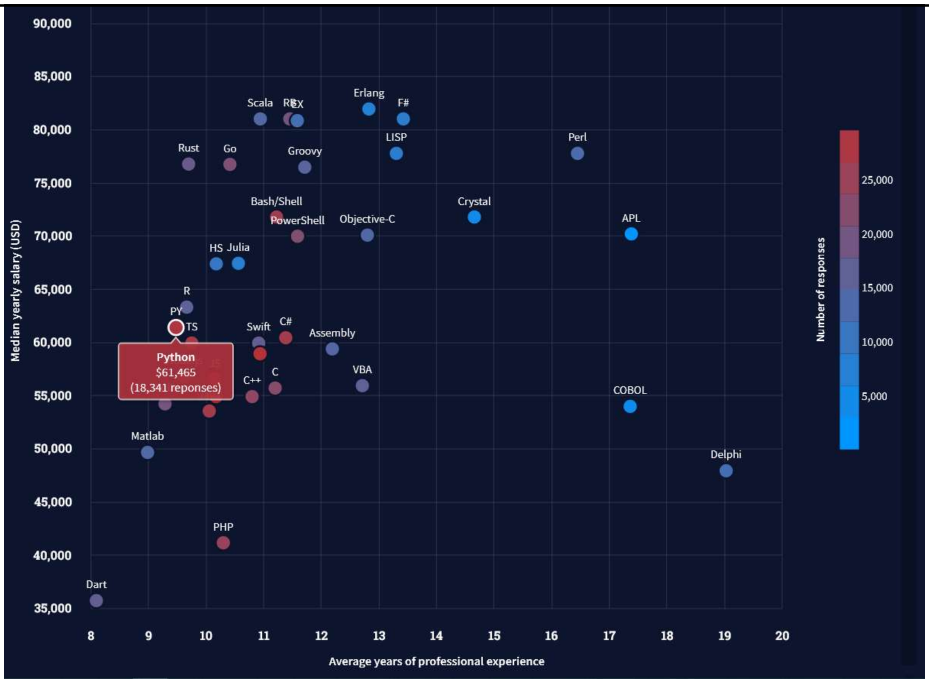
Worldwide, Aug 2022 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	28.11 %	-2.6 %
2		Java	17.35 %	-0.9 %
3		JavaScript	9.48 %	+0.2 %
4		C#	7.08 %	+0.1 %
5		C/C++	6.19 %	-0.3 %
6		PHP	5.47 %	-0.8 %
7		R	4.35 %	+0.6 %
8	↑↑	TypeScript	2.79 %	+1.1 %
9	↑↑	Swift	2.09 %	+0.5 %
10	↓↓	Objective-C	2.03 %	+0.2 %
11	↑	Go	2.03 %	+0.5 %
12	↓↓↓	Kotlin	1.78 %	-0.0 %

https://insights.stackoverflow.com/survey/2021#technology

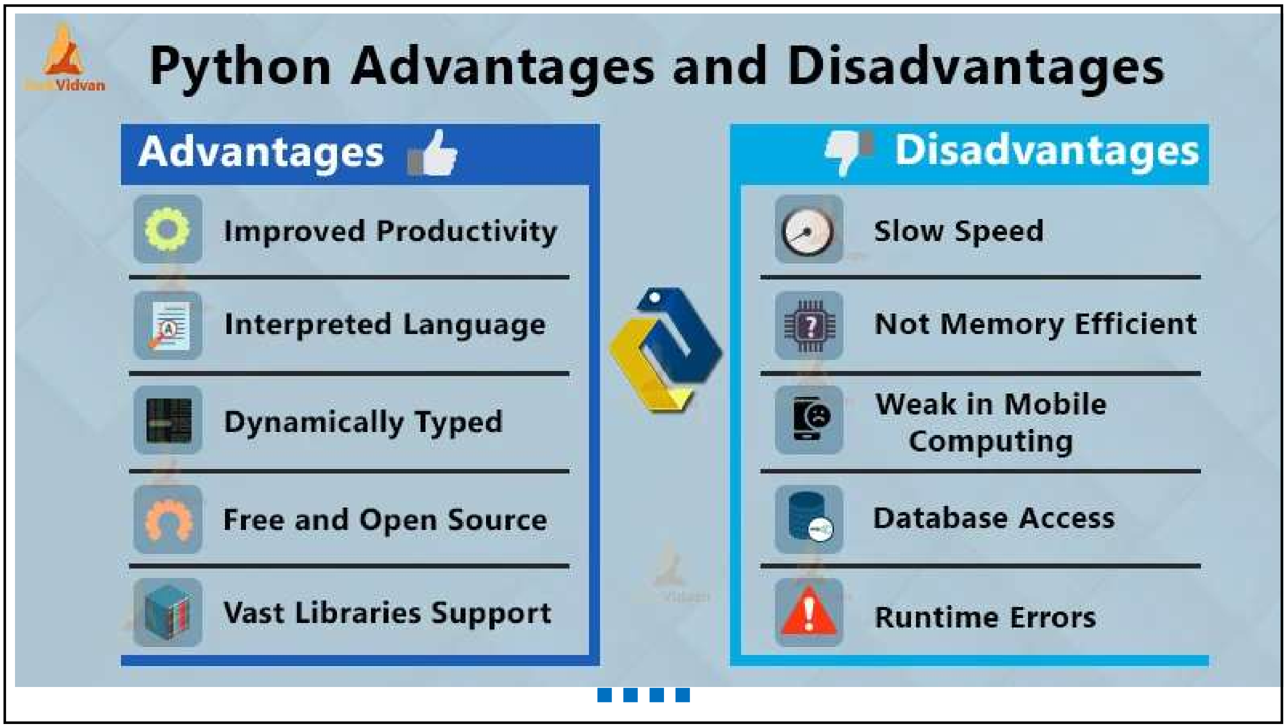
https://pypl.github.io/PYPL.html

9



https://insights.stackoverflow.com/survey/2021#work

10



Python Advantages and Disadvantages

Advantages

- Improved Productivity
- Interpreted Language
- Dynamically Typed
- Free and Open Source
- Vast Libraries Support

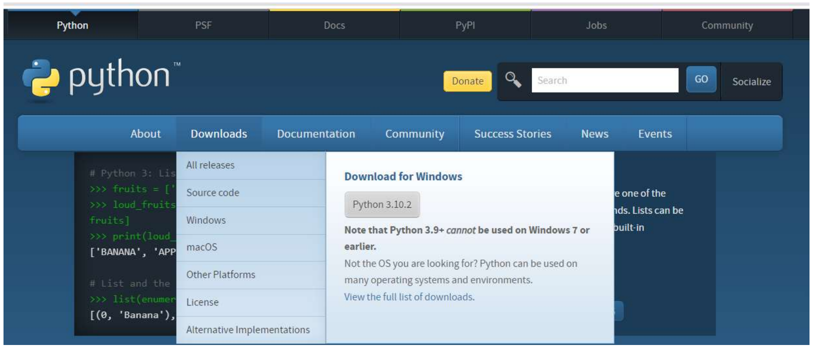
Disadvantages

- Slow Speed
- Not Memory Efficient
- Weak in Mobile Computing
- Database Access
- Runtime Errors

11

Using Python

- Google Colab: https://colab.research.google.com/
- Install on computer
  - https://www.python.org/downloads



```
# Python 3: Lis
>>> fruits = ['
>>> loud_fruits
fruits]
>>> print(loud_
['BANANA', 'APP

# List and the
>>> list(enumer
[(0, 'Banana'),
```

Download for Windows

Python 3.10.2

Note that Python 3.9+ *cannot* be used on Windows 7 or earlier.

Not the OS you are looking for? Python can be used on many operating systems and environments.

View the full list of downloads.

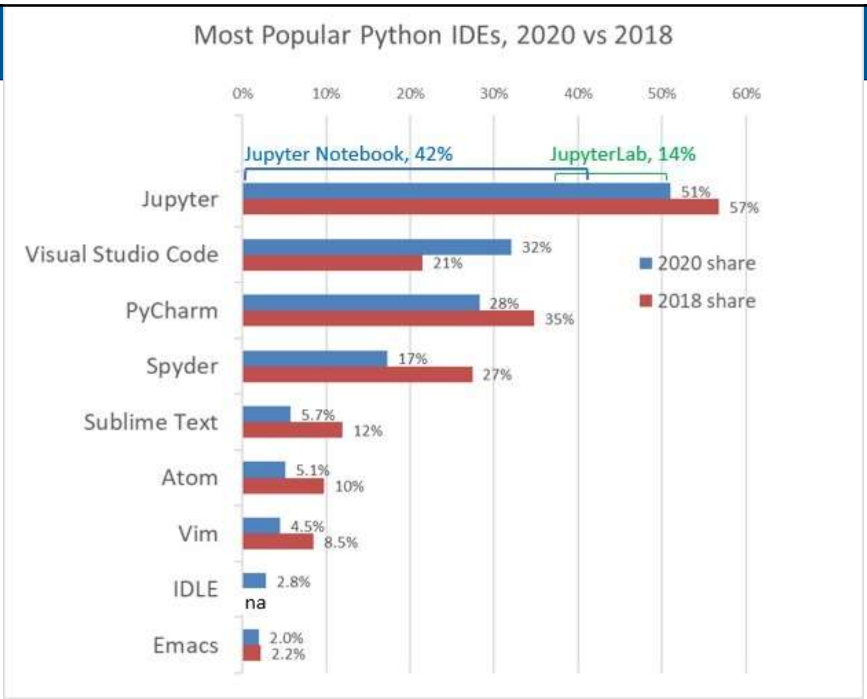
e one of the
nds. Lists can be
built-in

12

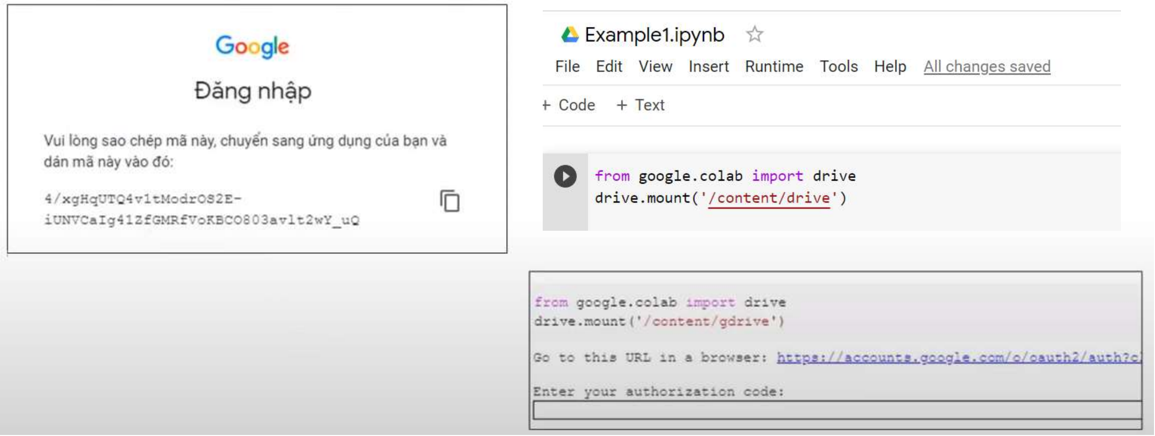
12

Best Python IDE

PC
Visual Studio Code
Sublime Text
Vim
GNU Emacs
SPYDER
Spyder
Atom
jupyter
Jupyter
Eclipse
IJ
IntelliJ IDEA
Notepad++
Notepad++
InterviewBit



Google Colab



Google

Đăng nhập

Vui lòng sao chép mã này, chuyển sang ứng dụng của bạn và dán mã này vào đó:

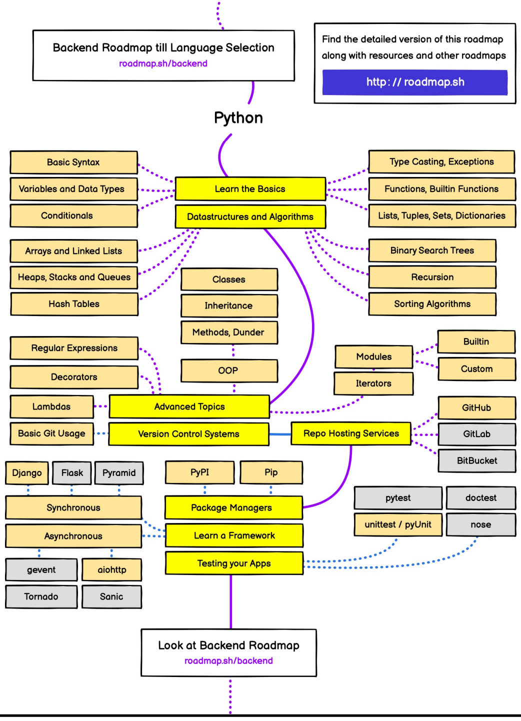
```
4/xgHqUTQ4v1tModrOS2E-
iUNVCaIg41ZfGMRfVoKBCO8O3av1t2wY_uQ
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
from google.colab import drive
drive.mount('/content/gdrive')

Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?c

Enter your authorization code:
```

15

15



Backend Roadmap till Language Selection
roadmap.sh/backend

Find the detailed version of this roadmap along with resources and other roadmaps
http://roadmap.sh

Python

Learn the Basics: Basic Syntax, Variables and Data Types, Conditionals, Type Casting, Exceptions, Functions, Builtin Functions, Lists, Tuples, Sets, Dictionaries

Datastructures and Algorithms: Arrays and Linked Lists, Heaps, Stacks and Queues, Hash Tables, Binary Search Trees, Recursion, Sorting Algorithms

OOP: Classes, Inheritance, Methods, Dunder

Modules: Builtin, Custom

Iterators

Advanced Topics: Regular Expressions, Decorators, Lambdas

Version Control Systems: Basic Git Usage

Repo Hosting Services: GitHub, GitLab, BitBucket

Package Managers: PyPI, Pip

Learn a Framework: Synchronous (Django, Flask, Pyramid), Asynchronous (gevent, aiohttp, Tornado, Sanic)

Testing your Apps: pytest, doctest, unittest / pyUnit, nose

Look at Backend Roadmap
roadmap.sh/backend

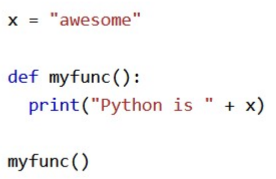
16

Python Variables

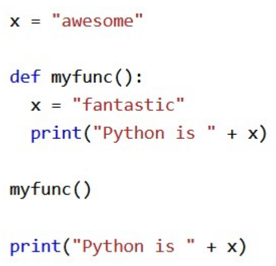
- Variables are containers for storing data values
- Python has no command for declaring a variable
- A variable is created the moment you first assign a value to it
- Variables do not need to be declared with any particular *type*, and can even change type after they have been set
- Variable names are case-sensitive
  - **Camel Case**
  - **Pascal Case**
  - **Snake Case**

Global Variables

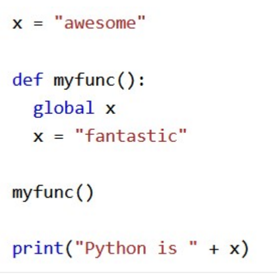
-



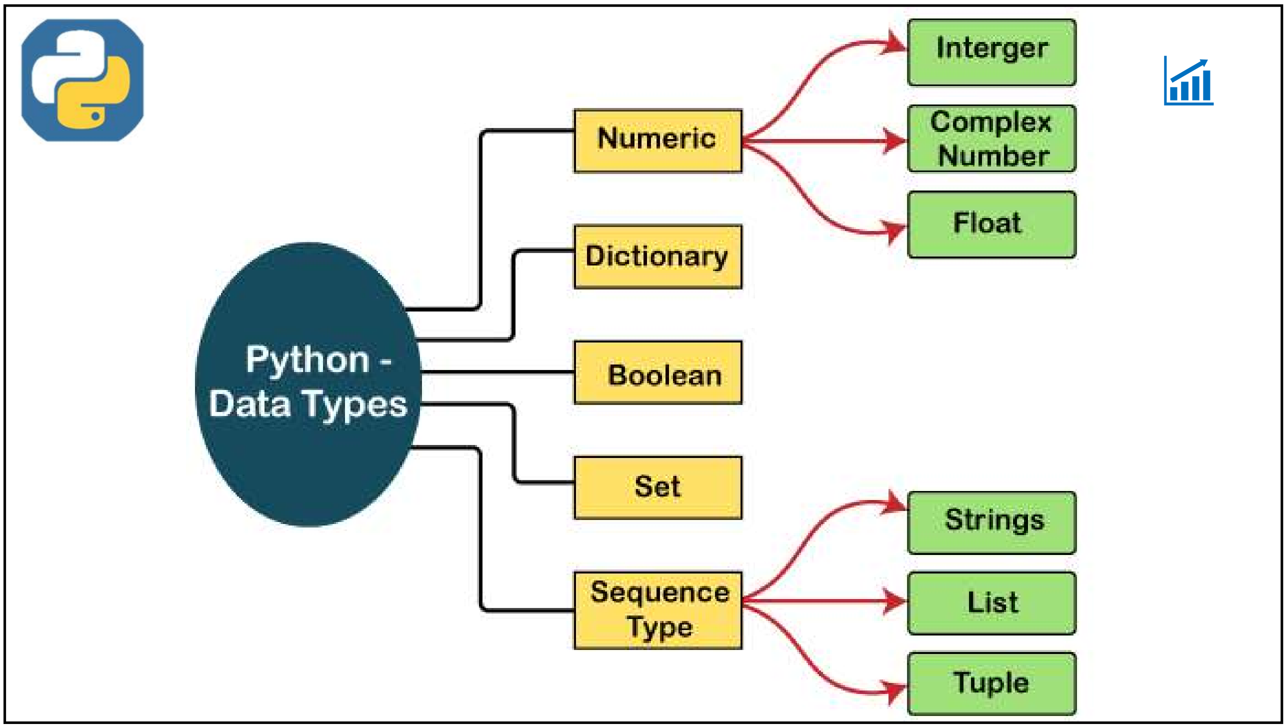
```
x = "awesome"

def myfunc():
  print("Python is " + x)

myfunc()
```



```
x = "awesome"

def myfunc():
  x = "fantastic"
  print("Python is " + x)

myfunc()

print("Python is " + x)
```



```
x = "awesome"

def myfunc():
  global x
  x = "fantastic"

myfunc()

print("Python is " + x)
```



Python - Data Types

Numeric

Interger

Complex Number

Float

Dictionary

Boolean

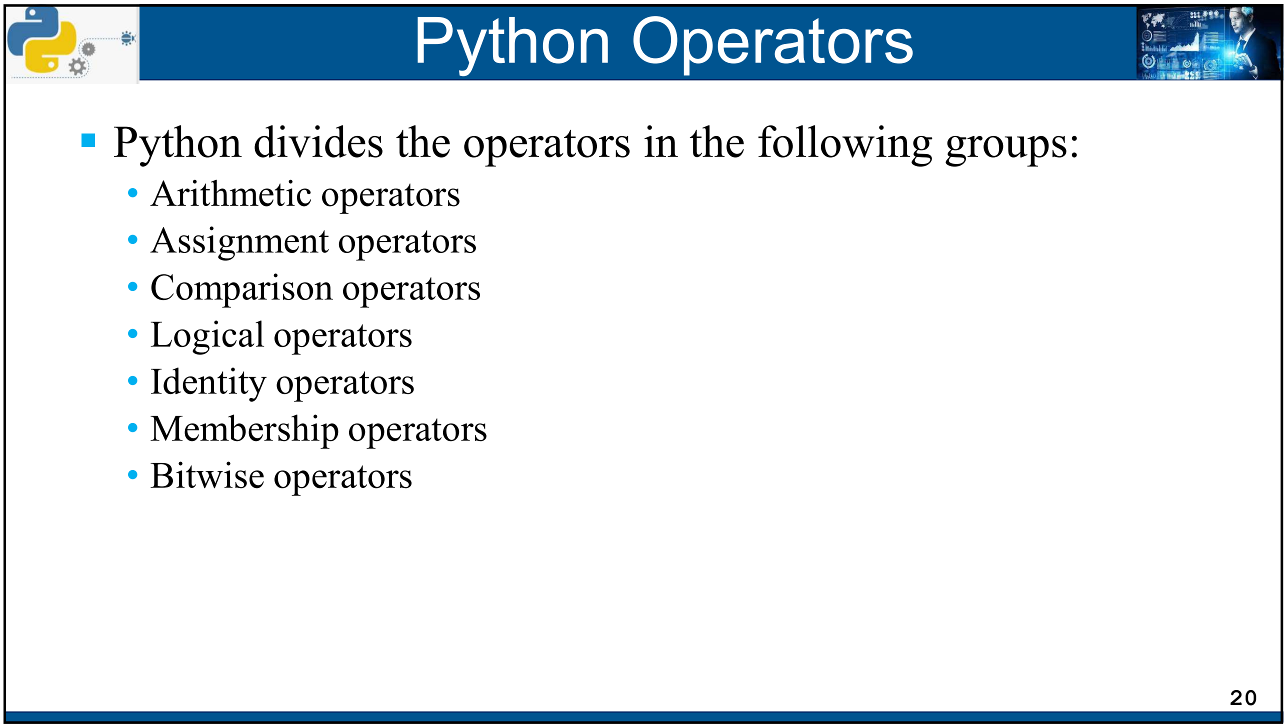
Set

Sequence Type

Strings

List

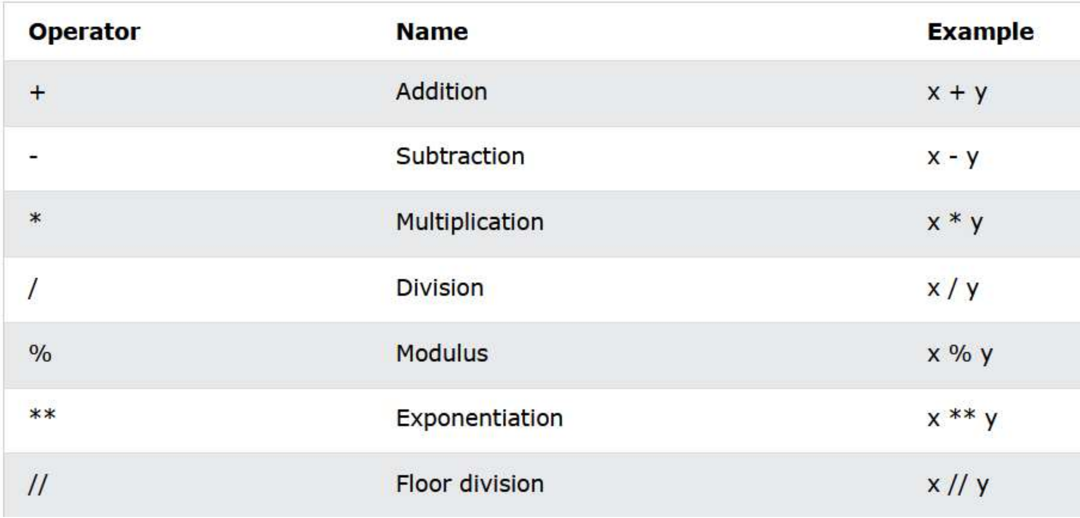
Tuple



Python Operators

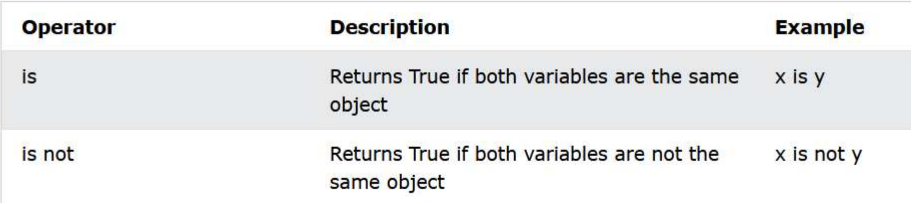
- Python divides the operators in the following groups:
  - Arithmetic operators
  - Assignment operators
  - Comparison operators
  - Logical operators
  - Identity operators
  - Membership operators
  - Bitwise operators

Arithmetic operators



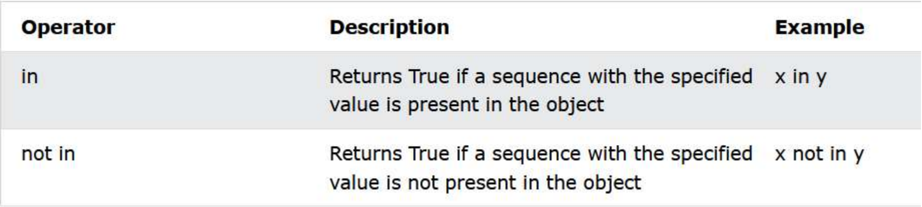
Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

Identity operators



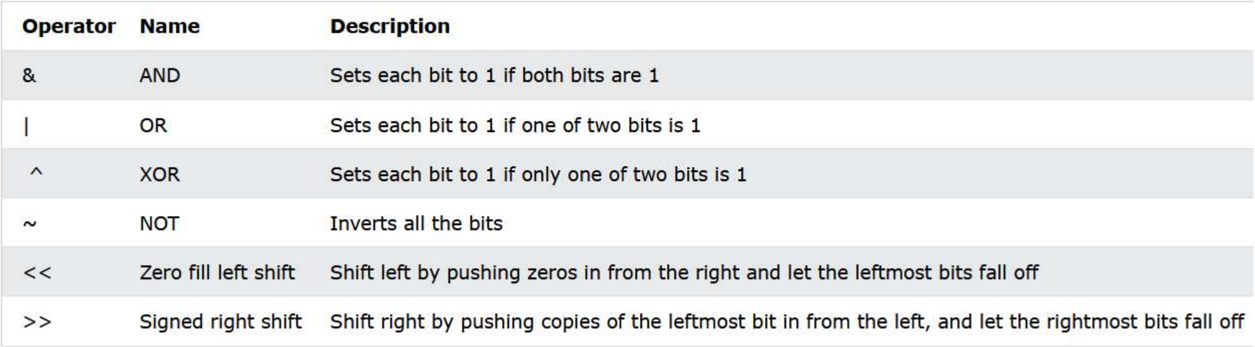
Operator	Description	Example
is	Returns True if both variables are the same object	x is y
is not	Returns True if both variables are not the same object	x is not y

Membership operators



Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns True if a sequence with the specified value is not present in the object	x not in y

Bitwise operators

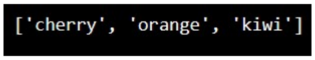


Operator	Name	Description
&	AND	Sets each bit to 1 if both bits are 1
|	OR	Sets each bit to 1 if one of two bits is 1
^	XOR	Sets each bit to 1 if only one of two bits is 1
~	NOT	Inverts all the bits
<<	Zero fill left shift	Shift left by pushing zeros in from the right and let the leftmost bits fall off
>>	Signed right shift	Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off

Lists

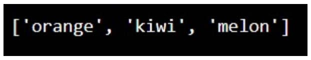
- thislist = ["apple", "banana", "cherry"]
print(thislist)

- thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[2:5])



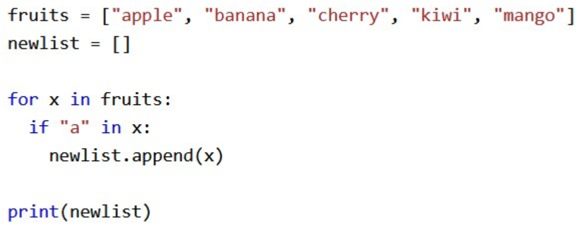
```
['cherry', 'orange', 'kiwi']
```

- thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[-4:-1])

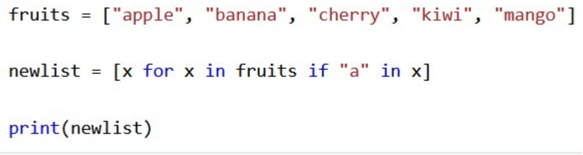


```
['orange', 'kiwi', 'melon']
```

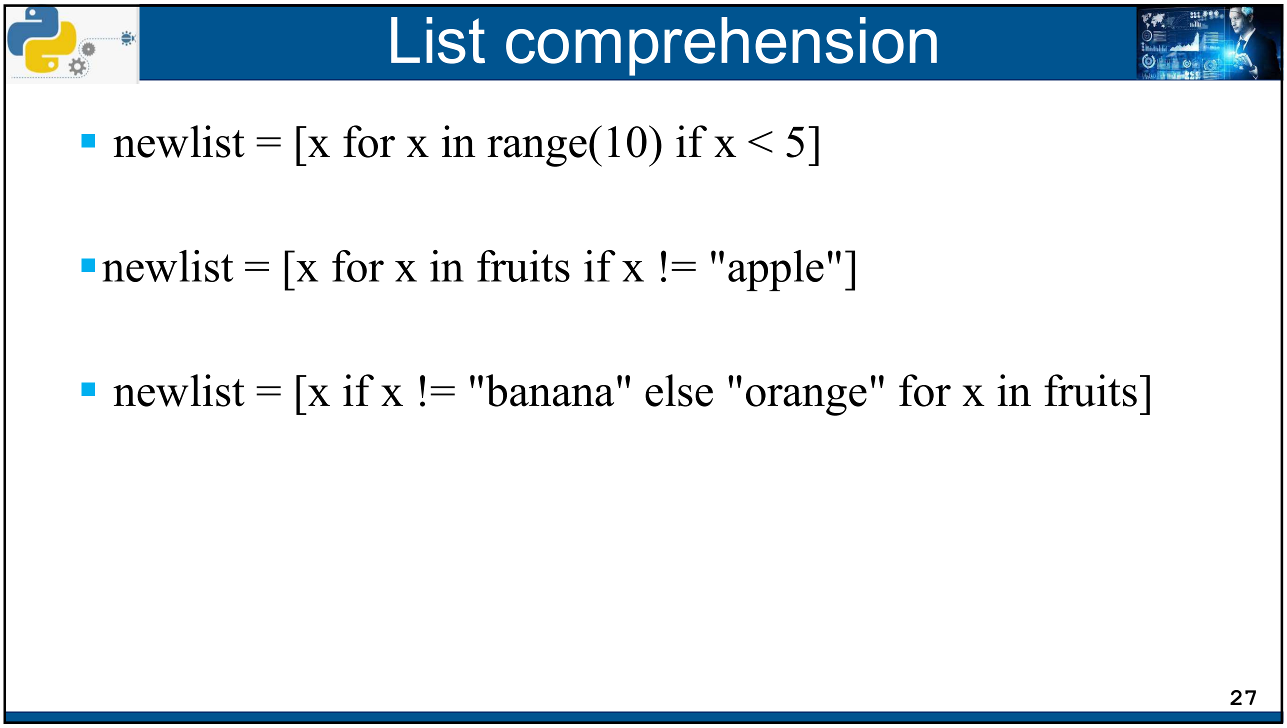
List

- List comprehension
  - List comprehension offers a shorter syntax when you want to create a new list based on the values of an existing list
  - 

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]
newlist = []

for x in fruits:
  if "a" in x:
    newlist.append(x)

print(newlist)
```



```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

newlist = [x for x in fruits if "a" in x]

print(newlist)
```

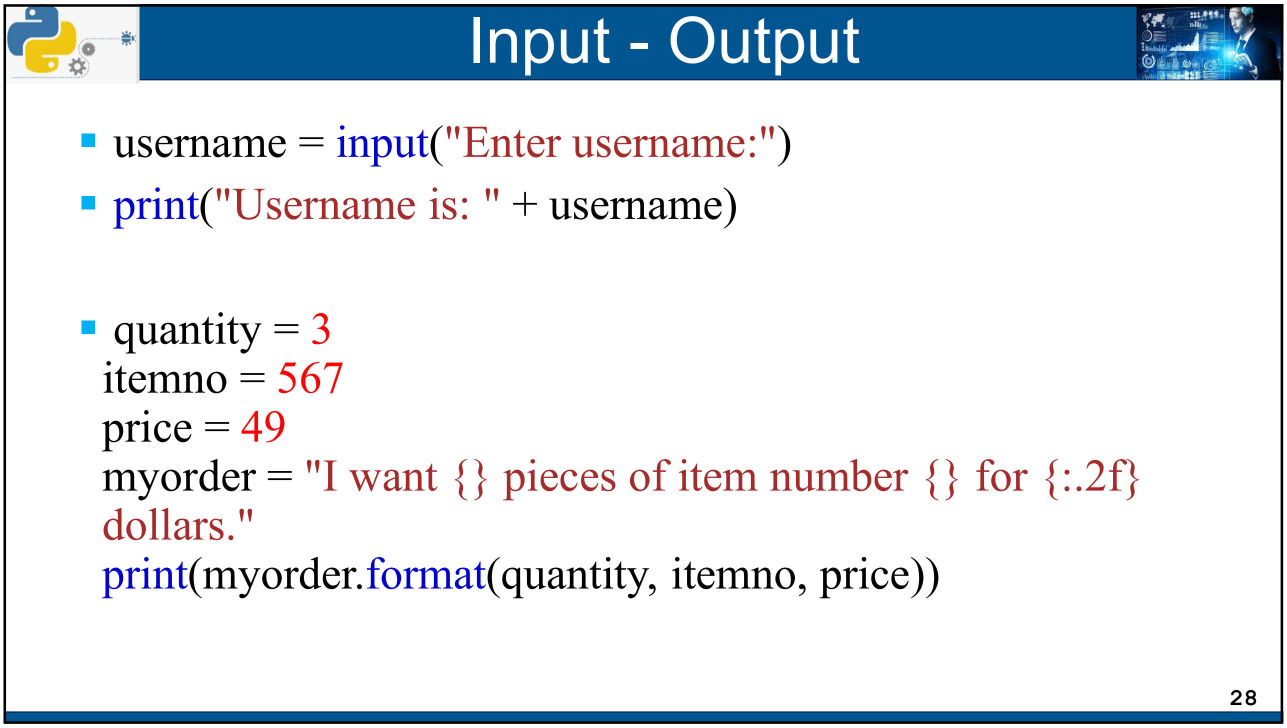


List comprehension

- `newlist = [x for x in range(10) if x < 5]`

- `newlist = [x for x in fruits if x != "apple"]`

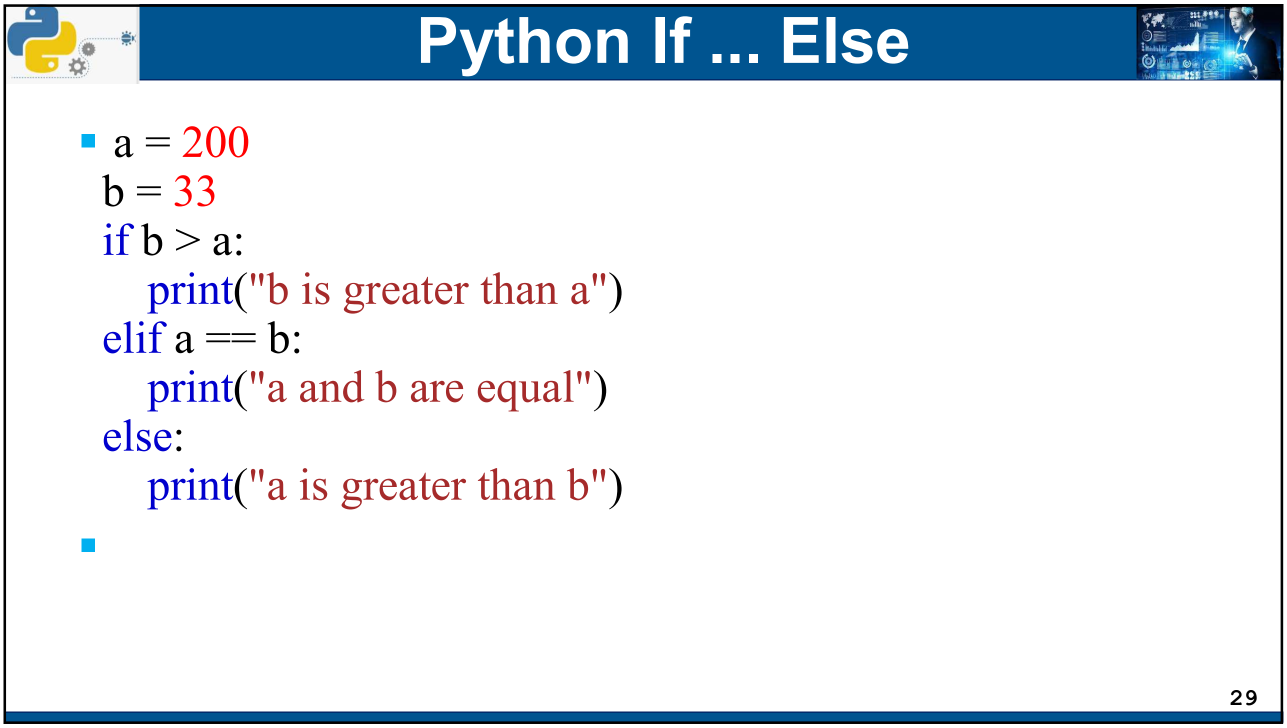
- `newlist = [x if x != "banana" else "orange" for x in fruits]`



Input - Output

- `username = input("Enter username:")`
- `print("Username is: " + username)`

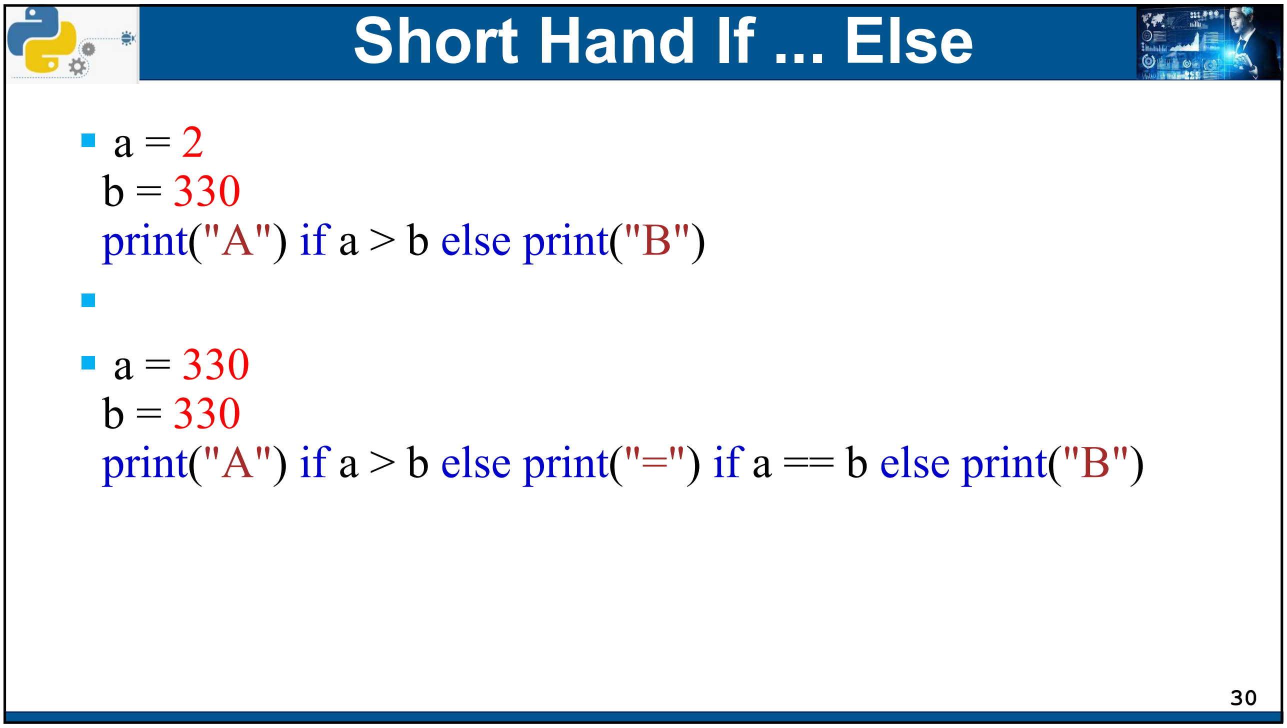
-
  ```
  quantity = 3
  itemno = 567
  price = 49
  myorder = "I want {} pieces of item number {} for {:.2f} dollars."
  print(myorder.format(quantity, itemno, price))
  ```



Python If ... Else

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

-

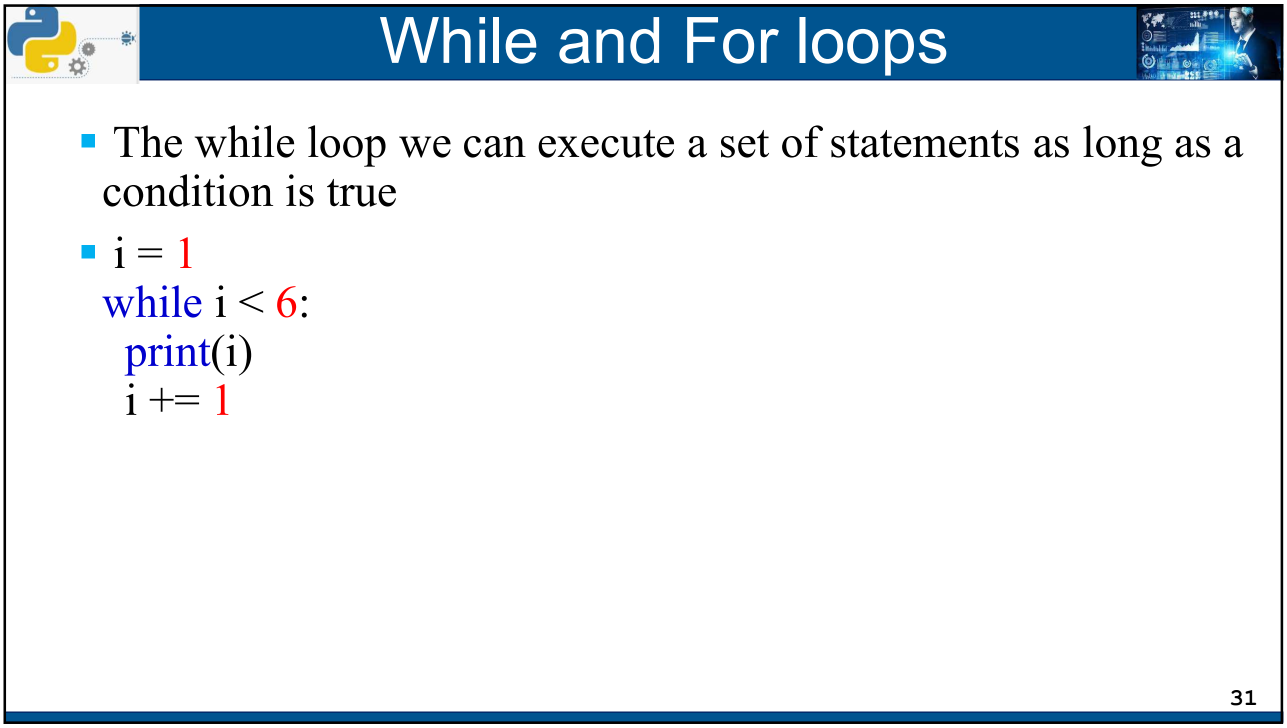


Short Hand If ... Else

```
a = 2
b = 330
print("A") if a > b else print("B")
```

-

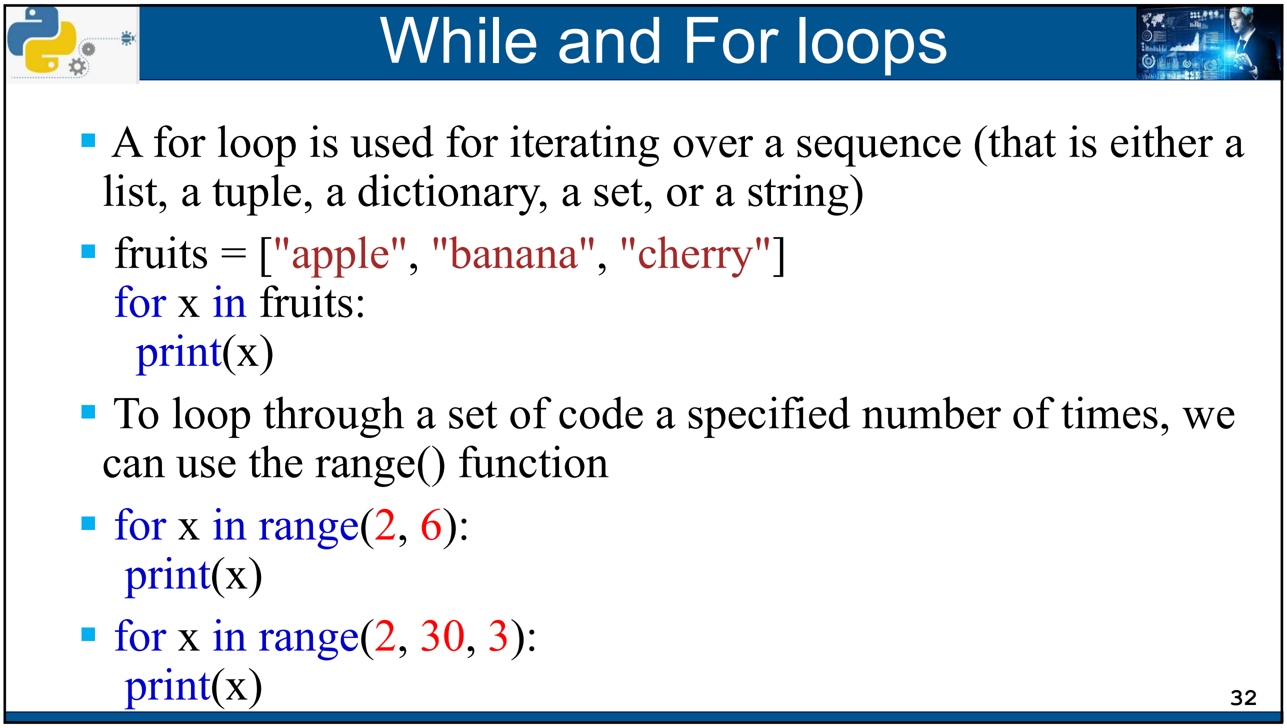
```
a = 330
b = 330
print("A") if a > b else print("=") if a == b else print("B")
```



While and For loops

- The while loop we can execute a set of statements as long as a condition is true
-
```
i = 1
while i < 6:
  print(i)
  i += 1
```

31



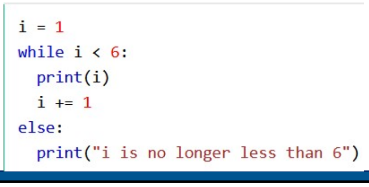
While and For loops

- A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string)
-
```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```
- To loop through a set of code a specified number of times, we can use the range() function
-
```
for x in range(2, 6):
 print(x)
```
-
```
for x in range(2, 30, 3):
 print(x)
```

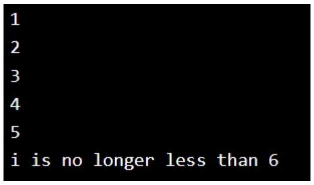
32

While and For loops

- break Statement
  - With the break statement we can stop the loop even if the while condition is true
- continue Statement
  - With the continue statement we can stop the current iteration, and continue with the next
- else in for/while loop
  - With the else statement we can run a block of code once when the condition no longer is true



```
i = 1
while i < 6:
  print(i)
  i += 1
else:
  print("i is no longer less than 6")
```



```
1
2
3
4
5
i is no longer less than 6
```

Exercise

- Tính tổng n số nguyên đầu tiên
- In danh sách các số chẵn thuộc (0,n]
- Tính tổng các số lẻ <=n, trừ các số chia hết cho 3
- Tính trung bình cộng của n số nguyên dương được nhập từ bàn phím. Nhập sai 1 số (nhập số âm) thoát chương trình

Question?