wtech
Teknolojide Kadın Derneği

Introduction to Machine Learning with Python

Introduction, Variables and `for` Loops

Dr. Süha Tuna
İTÜ Informatics Institute

CONTENT

Why Python?

- **Easy to Learn and Readable**
  - Simple syntax, similar to English. Emphasizes readability and reduces the cost of program maintenance.
- **Interpreted Language**
  - Code is executed line by line at runtime (no need to compile). Makes debugging easier.
- **Dynamically Typed**
  - No need to declare variable types explicitly. Variable types are inferred at runtime.
- **High-Level Language**
  - Abstracts low-level operations (like memory management). Lets you focus on logic and functionality.

Why Python?

- **Object-Oriented**
  - Supports classes, inheritance, encapsulation, and polymorphism. Everything is an object in Python.
- **Multi-Paradigm**
  - Supports procedural, object-oriented, and functional programming.
- **Extensive Standard Library**
  - Includes libraries for file I/O, regex, networking, math, web services, etc.
- **Cross-Platform**
  - Python code runs on Windows, Linux, macOS, etc. without modification.

Why Python?

- **Large Community and Ecosystem**
  - Rich set of third-party libraries and massive community support.
- **Embeddable and Extensible**
  - Can embed Python into C/C++ and call C/C++ functions using libraries like ctypes.
- **Garbage Collection**
  - Automatic memory management with reference counting and cyclic garbage collector.
- **Interactive Mode**
  - Python shell lets you test code snippets quickly; useful for prototyping and debugging.

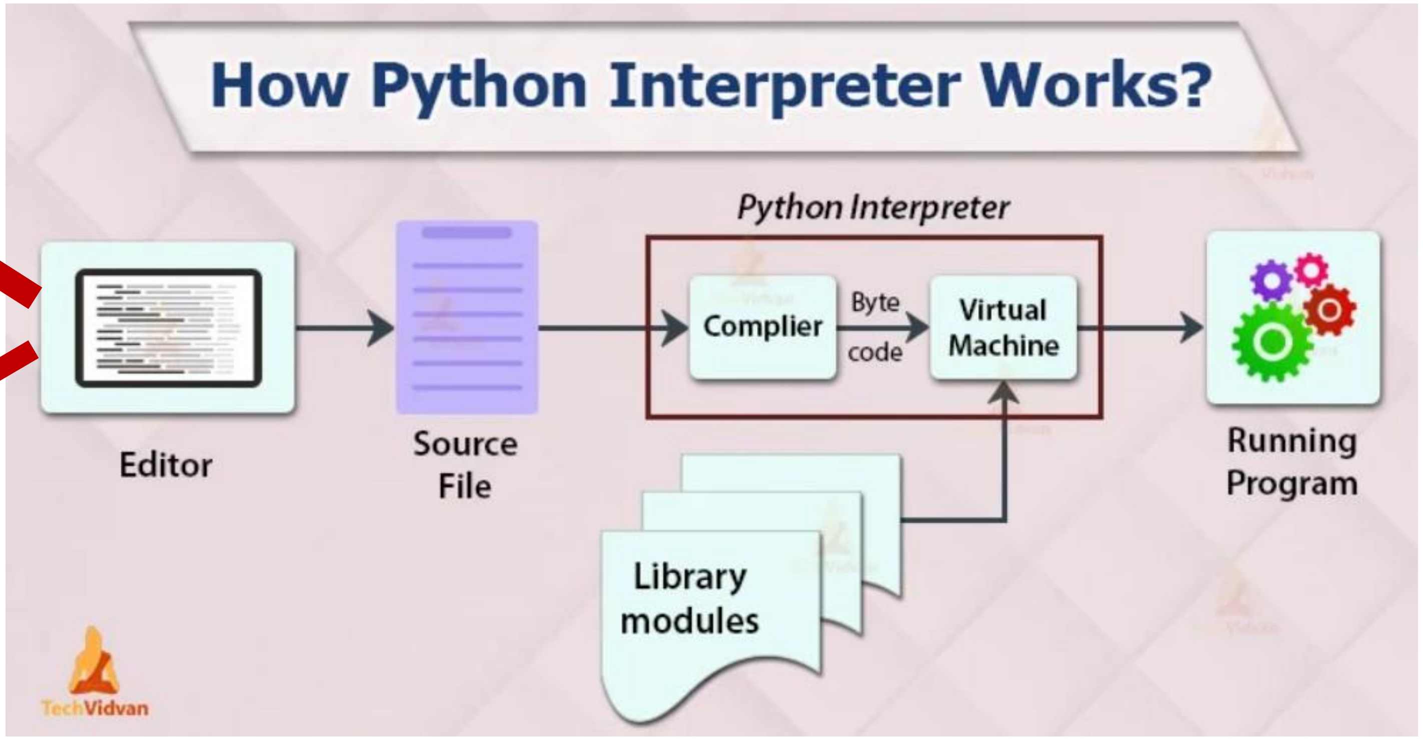
Why Python?

- **Open Source**
  - Free to use and distribute. Developed under an OSI-approved open-source license.
- **Strong Support for Integration**
  - Interfaces with C, C++, Java, and .NET; useful in enterprise and embedded systems.
- **Supports Exception Handling**
  - Structured error handling using try, except, finally.
- **Case sensitive**
  - `Print`, `print` and `PRINT` are different!
- **Indentation is important**
  - Indents are employed to create programming block

How to write a Python program?

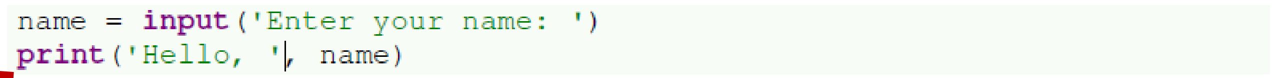
ANACONDA

python + colab

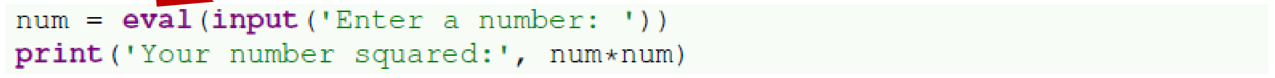


Getting input

```
variable name = input(message to user)
```



```
name = input('Enter your name: ')
print('Hello, ', name)
```



```
num = eval(input('Enter a number: '))
print('Your number squared:', num*num)
```

The **`eval`** function converts the text entered by the user into a number. One nice feature of this is you can enter expressions, like `3*12+5`, and **`eval`** will compute them for you.

First programs, typing things in

```
temp = eval(input('Enter a temperature in Celsius: '))
print('In Fahrenheit, that is', 9/5*temp+32)
```

```
num1 = eval(input('Enter the first number: '))
num2 = eval(input('Enter the second number: '))
print('The average of the numbers you entered is', (num1+num2)/2)
```

Printing

```
print('Hi there')
```

```
print('3+4')
print(3+4)
```

```
print('The value of 3+4 is', 3+4)
print('A', 1, 'XYZ', 2)
```

```
print ('The value of 3+4 is', 3+4, '.')
print ('The value of 3+4 is ', 3+4, '.', sep='')
```

```
print('On the first line')
print('On the second line')
```

```
print('On the first line', end='')
print('On the second line')
```

Variables

```
temp = eval(input('Enter a temperature in Celsius: '))
print('In Fahrenheit, that is', 9/5*temp+32)
```

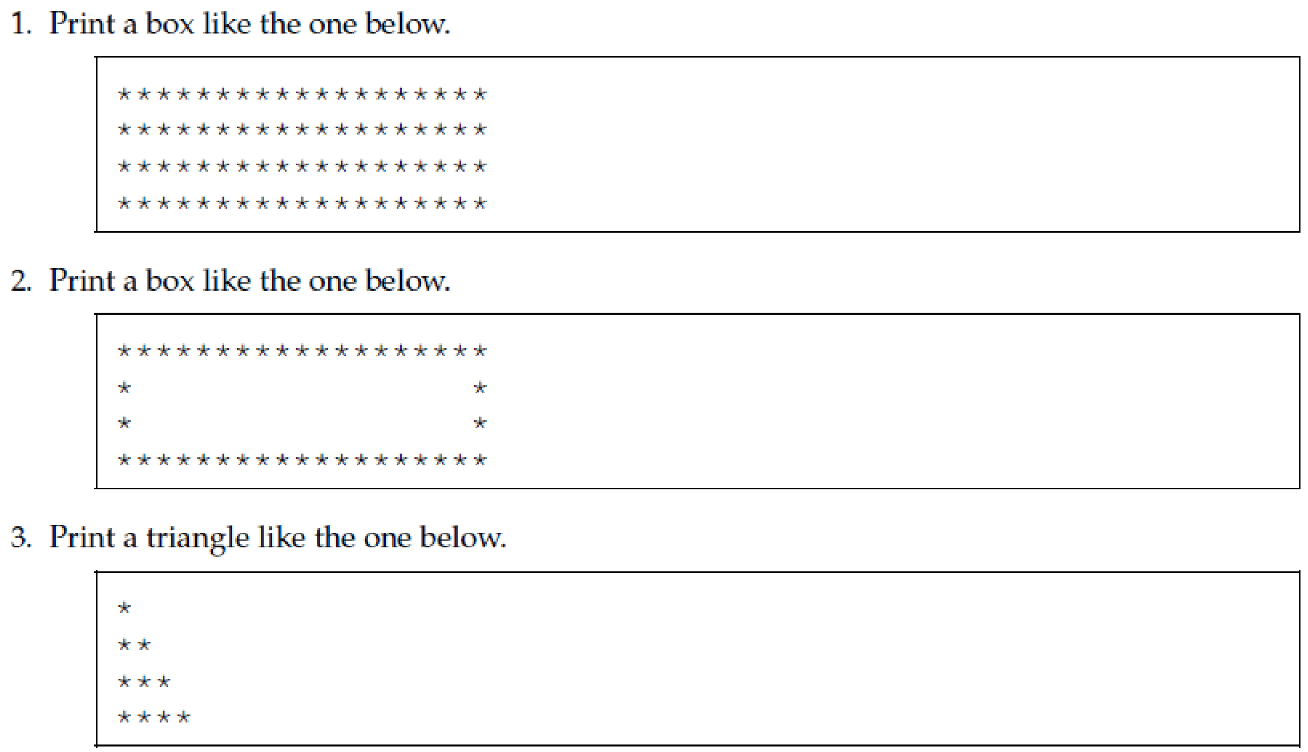
```
temp = eval(input('Enter a temperature in Celsius: '))
f_temp = 9/5*temp+32
print('In Fahrenheit, that is', f_temp)
if f_temp > 212:
    print('That temperature is above the boiling point.')
if f_temp < 32:
    print('That temperature is below the freezing point.')
```

```
x=3
y=4
z=x+y
z=z+1
x=y
y=5
```

Variable names

- Variable names can contain letters, numbers, and the underscore.
- Variable names cannot contain spaces.
- Variable names cannot start with a number.
- Case matters—for instance, `temp` and `Temp` are different.

Exercises



1. Print a box like the one below.

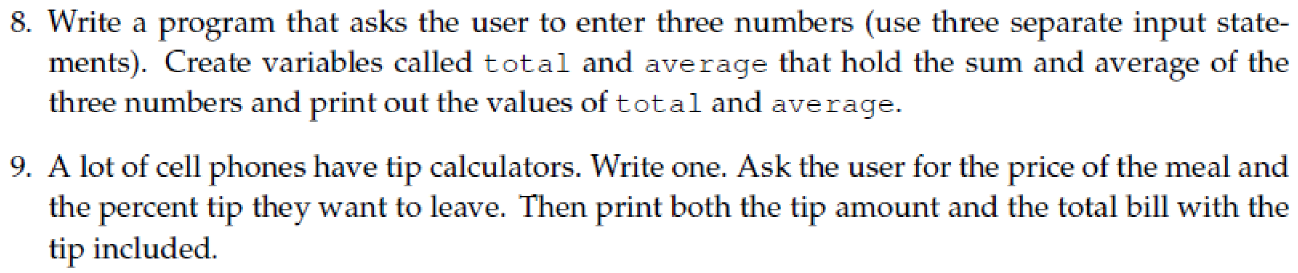
```
******************
******************
******************
******************
```

2. Print a box like the one below.

```
******************
*                *
*                *
******************
```

3. Print a triangle like the one below.

```
*
**
***
****
```



8. Write a program that asks the user to enter three numbers (use three separate input statements). Create variables called `total` and `average` that hold the sum and average of the three numbers and print out the values of `total` and `average`.

9. A lot of cell phones have tip calculators. Write one. Ask the user for the price of the meal and the percent tip they want to leave. Then print both the tip amount and the total bill with the tip included.

for loops

```
for i in range(10):
    print('Hello')
```

```
for i in range(3):
    num = eval(input('Enter a number: '))
    print ('The square of your number is', num*num)
print('The loop is now done.')
```

```
print('A')
print('B')
for i in range(5):
    print('C')
    print('D')
print('E')
```

```
print('A')
print('B')
for i in range(5):
    print('C')
for i in range(5):
    print('D')
print('E')
```

The loop variable, range function

```
for i in range(100):
    print(i)
```

```
for i in range(3):
    print(i+1, '-- Hello')
```

```
for i in range(100):                for wacky_name in range(100):
    print(i)                            print(wacky_name)
```

Statement	Values generated
`range(10)`	0,1,2,3,4,5,6,7,8,9
`range(1,10)`	1,2,3,4,5,6,7,8,9
`range(3,7)`	3,4,5,6
`range(2,15,3)`	2,5,8,11,14
`range(9,2,-1)`	9,8,7,6,5,4,3

```
for i in range(5,0,-1):
    print(i, end=' ')
print('Blast off!!')
```

A Trickier Example

```
for i in range(4):
    print('*'*6)
```

```
for i in range(4):
    print('*'*(i+1))
```

Exercises

1. Write a program that prints your name 100 times.

2. Write a program to fill the screen horizontally and vertically with your name. [Hint: add the option `end=''` into the `print` function to fill the screen horizontally.]

3. Write a program that outputs 100 lines, numbered 1 to 100, each with your name on it. The output should look like the output below.

```
1 Your name
2 Your name
3 Your name
4 Your name
...
100 Your name
```

4. Write a program that prints out a list of the integers from 1 to 20 and their squares. The output should look like this:

```
1 --- 1
2 --- 4
3 --- 9
...
20 --- 400
```

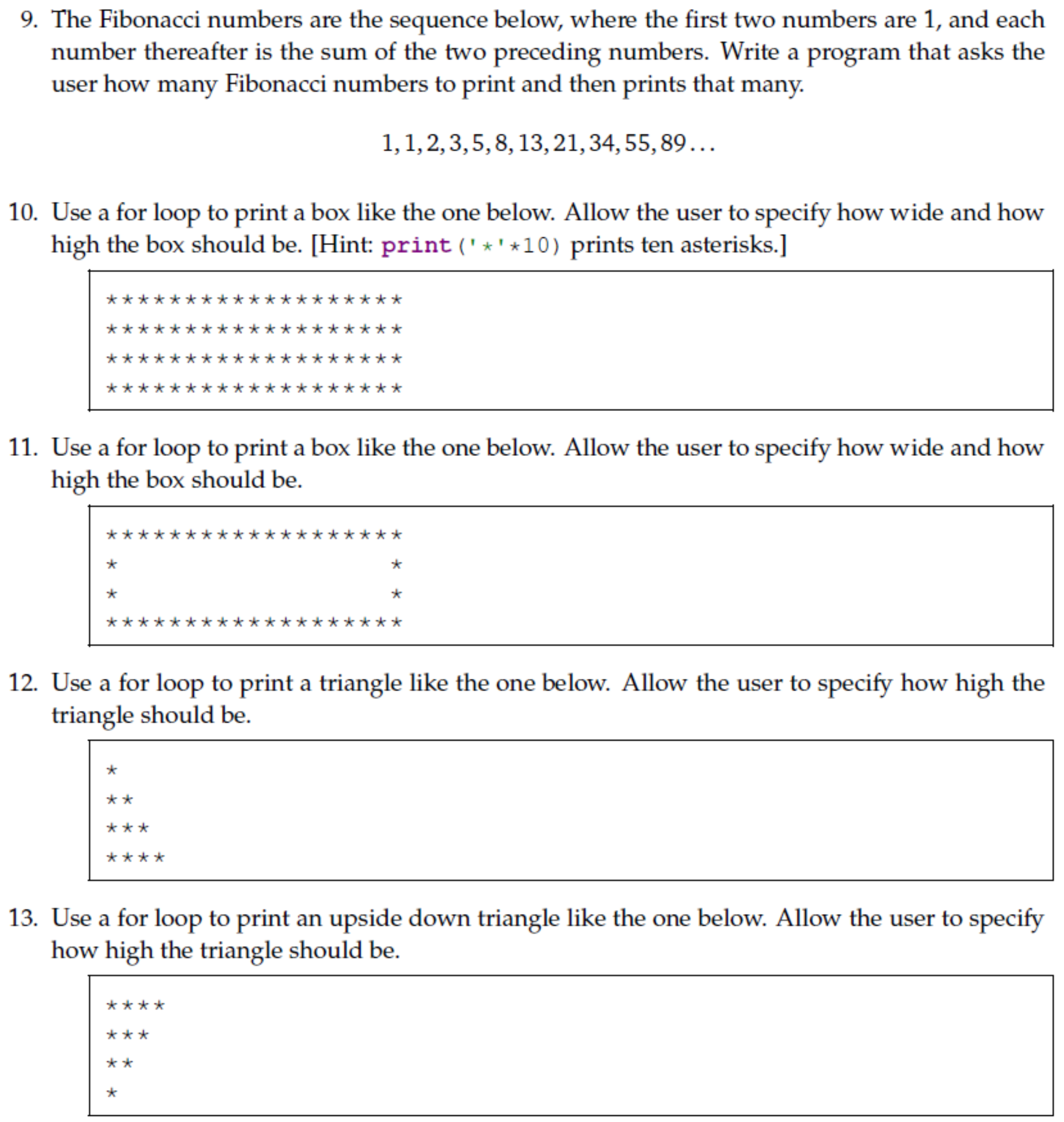
5. Write a program that uses a for loop to print the numbers 8, 11, 14, 17, 20, . . . , 83, 86, 89.

6. Write a program that uses a for loop to print the numbers 100, 98, 96, . . . , 4, 2.

7. Write a program that uses exactly four for loops to print the sequence of letters below.

```
AAAAAAAAAABBBBBBBCDCDCDCDEFFFFFFG
```

Exercises



9. The Fibonacci numbers are the sequence below, where the first two numbers are 1, and each number thereafter is the sum of the two preceding numbers. Write a program that asks the user how many Fibonacci numbers to print and then prints that many.

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89 \ldots$$

10. Use a for loop to print a box like the one below. Allow the user to specify how wide and how high the box should be. [Hint: `print('*'*10)` prints ten asterisks.]

```
*******************
*******************
*******************
*******************
```

11. Use a for loop to print a box like the one below. Allow the user to specify how wide and how high the box should be.

```
*******************
*                 *
*                 *
*******************
```

12. Use a for loop to print a triangle like the one below. Allow the user to specify how high the triangle should be.

```
*
**
***
****
```

13. Use a for loop to print an upside down triangle like the one below. Allow the user to specify how high the triangle should be.

```
****
***
**
*
```