

HKUST Future-Ready Scholars

Introduction to Game Programming using Python

Part 1: Number Guessing Game

20 April 2024



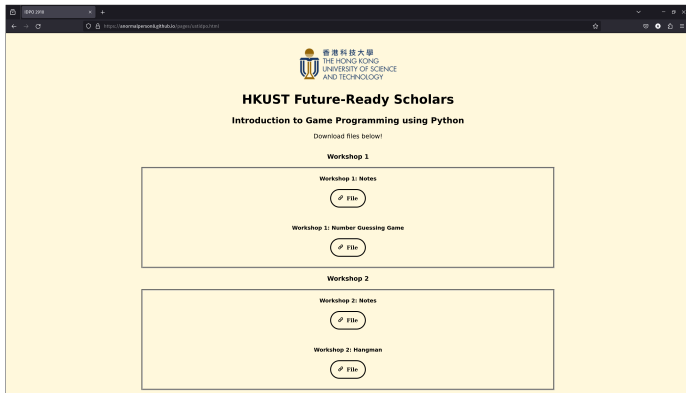
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

We will use Google Colab for the workshops.

<https://colab.research.google.com/>

You must have a Gmail account for it, create one if you do not.

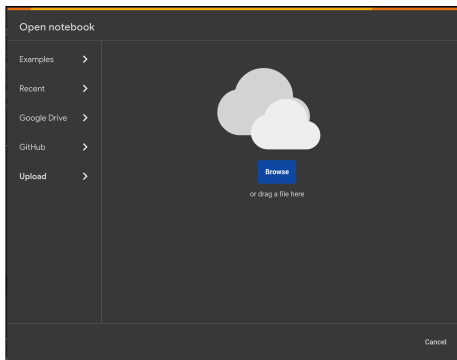
All materials today are at:
<https://bit.ly/ustidpo>



Download all files that belong to **Workshop 1** today.

Jupyter Notebook

Now upload your Jupyter Notebook file with **Files** → **Open Notebook**.



Upload the file **Number-Guessing.ipynb**.

Using Jupyter Notebook

You can type your code in these blocks. We call these blocks code cells.



```
print("Mum I am in HKUST typing code in a code block")
```

You can run a code cell with the button on the left.



```
print("Hello World!") # Prints "Hello World!"
```

World of Game Coding



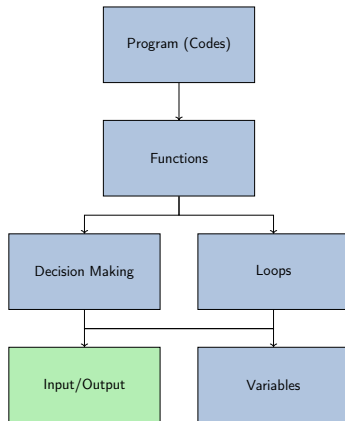
What is Python?

Did you know? Python was made by someone who was bored.
It's a language designed to be almost as understandable as English.
You will be using Python 3. Why? Because Python 1 and 2 are too old.



This is the logo of Python.

Contents



The first thing in Python - print() function

```
print("This is the print function.")
```

The first thing in Python - print() function

`print()` is a function that lets you print something, also known as text output.

```
print("Word") # This prints the word "Word".
```

Examples:

```
>>> print("Hello World")
```

Hello World

```
>>> print("Haha hehe")
```

Haha hehe

Printing multiple things

You can use a comma (,) to separate different things with a space.

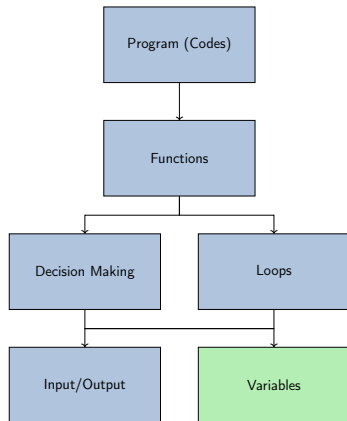
```
>>> print("Alpha", "Beta", "Gamma")
```

```
Alpha Beta Gamma
```

```
>>> print("Haha", "hehe")
```

```
Haha hehe
```

Contents



Imagine you borrow a box from the computer.



Give it a name and a value, you can now recall this value with the name!

Variables

The code usually goes:

```
variable_name = data
```

This means whatever data is, it is now stored in a variable with name variable_name.

Some basic variable types:

```
a = 5          # This is an integer (int) stored in a
b = True       # This is a boolean (bool) stored in b
c = 3.2        # This is a float (float) stored in c
d = "abc"      # This is a string (str) stored in d
e = 'abc'      # This is also a string stored in e
```

Variables - Integers

What are integers?

Integers are just like what you've learnt in Maths, numbers without decimal points. Are the following valid?

```
a = 5           # Valid
b = 12          # Valid
c = 69420       # Valid
d = -1984       # Valid
e = 32.5        # This would become a float instead
f = '5'         # This would become a string instead
```

Variables - Integer Arithmetic Operations

You can do normal operations on integers:

```
a = 1 + 2    # a stores the integer 3
b = 80 - 52  # b stores the integer 28
c = 69 * -2  # c stores the integer -138
d = 6 / 4    # d stores the float 1.5
e = 18 / 2   # e stores the float 9.0
```

Division in Python

Whether a number can be precisely divided or not, division returns a float.

Variables - Integer Arithmetic Operations

Operations with variables:

```
a = 100
```

```
b = 12
```

```
c = a + b    # c stores the integer 112
```

```
d = b - a    # d stores the integer -88
```

```
e = a * -b    # e stores the integer -1200
```

```
f = a / b     # f stores the float 8.333333333333334
```

Variables - Integer Arithmetic Operations

Then how do we get an integer output?

```
a = 100
```

```
b = 12
```

```
c = a // b  # c stores the integer 8  
            # // operator takes the closest and smaller  
            # integer from the division operation  
d = a % b   # d stores the integer 4  
            # % operator takes the remainder of a  
            # division operation
```

Variables - Integer Arithmetic Operations

Also, the power (exponent) operation:

```
a = 2
```

```
b = 5
```

```
c = a ** b  # c stores the integer 32  
            # ** operator means power
```

Variables - Floats

What are floats?

Floats are numbers with decimal points.

Arithmetic operators we learnt can be applied as well.

```
a = 0.2      # a stores the float 0.2
b = 3.0      # b stores the float 3.0
c = a + b    # c stores the float 3.2
d = b / a    # d stores the float 15.0
e = a ** b   # e stores the float 0.0080000000000000002
```

Inaccuracies

Inaccuracies happen with decimals in Python. Be careful when dealing with floats.

Variables - Floats

What happens when you combine floats and integers?

```
a = 0.2      # a stores the float 0.2
b = 3        # b stores the integer 3
c = a + b    # c stores the float 3.2
d = b / a    # d stores the float 15.0
e = a ** b   # e stores the float 0.0080000000000000002
```

Arithmetic operations between int and float

Arithmetic operations between integers and floats converts the integer into a float first before operating.

Variables - Boolean values

What are boolean values?

There are only 2 boolean values in existence: True and False.

```
a = True
```

```
b = False
```

We will elaborate more on boolean values later.

What are strings?

```
a = "word"    # a stores the string "word"
b = 'word2'   # b stores the string "word2"
c = '5.20'    # c stores the string "5.20"
d = 'abc"     # error
```

Quotes

In Python you must use corresponding quotation marks for strings.

Variables - Strings

How do I put the symbols ' and " into a string?

For ":

```
a = "word\" # a stores the string "word"  
b = 'word"' # b stores the same string as a
```

Same goes for single quotes ':

```
a = 'word\'' # a stores the string "word"  
b = "word'" # b stores the same string as a
```


There are additional symbols in strings.

```
a = "word\n" # \n represents the newline character  
b = "word\t" # \t represents the tab character
```

Variables - Strings

Example:

```
a = "haha"
```

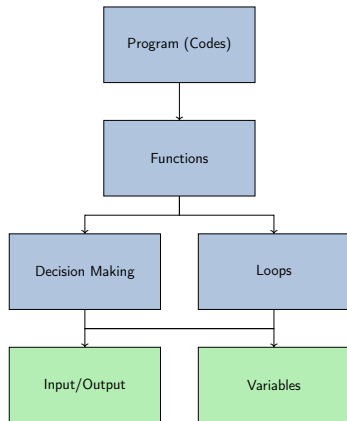
```
b = "hehe"
```

```
c = a + b      # c stores the string "hahahehe"
```

Concatenation of strings

You can concatenate (add) strings together with the addition symbol.

Contents



Variables in output - Revisiting the `print()` function

How do we print variables?

```
a = 5
print(a)          # 5
b = "haha"
print(b)          # haha
print(a + 2)      # 7
print(b + "a")    # hahaa
```

Calculation

We can calculate expressions inside the `print()` function.

Variables in output - Revisiting the `print()` function

How do we print variables?

```
a = 5
print(a)          # 5
b = "haha"
print(a, b)       # 5 haha
print(b, b)       # haha haha
```

The comma

Using `,` in `print()` would add a space in between the 2 items.

Variables in output - Revisiting the print() function

How do we print variables?

```
a = 5
print(a)          # 5
b = "haha"
print(b)          # haha
print(a + "5")    # error
print(b + 2)      # error
print(a + b)      # error
```

Addition

You cannot use addition to print things of incompatible types.
int and float types are not incompatible because all int are converted to float if needed during operation.

Variables in output - Revisiting the `print()` function

How do we print variables?

```
a = 5
```

```
b = 32
```

```
c = 32.0
```

```
print(a * b)      # 160
```

```
print(a * c)      # 160.0
```

Takeaway

`print()` function evaluates the expression inside the brackets first before actually printing.

Task 1

Now go to your Number Guessing Game file's task 1.

All lines you have to do is marked with “# TODO:”.
Try to finish the `print()` statements!

The End
Made in \LaTeX
Last updated: 29 Mar 2024

Additional content

Here are some additional content that we didn't have time to mention in the workshop.

More on print() function

In Python, the `print()` function automatically adds a new line after execution. We, however, can stop that.

The `end=` tag allows us to define the character added when `print()` is executed.

```
print(5, end="")  
print(4)  
print("a", end="abc")  
print("d", end=" ")  
print("e")  
# What is the output?  
# Output: 54  
#          aabcd e
```

End of line

Remember to include a new line `\n` in the last line of a printed string. Else it may mess up the future outputs from other lines of the code or the computer terminal.

More on print() function

We mentioned that whenever `,` is used in `print()`, the items would be separated by a space.

This can actually be changed using the `sep=` tag.

```
>>> print("100", 100, end="\n3\n")
```

```
>>> 100 100
```

```
3
```

```
>>> print("100", 100, sep="a", end="\n3\n")
```

```
>>> 100a100
```

```
3
```

More on print() function

Another example:

```
>>> a = 5
>>> b = 10
>>> print(a, b, a + b, end="20\n")
>>> 5 10 1520
>>> print(a, b, a + b, sep="", end="20\n")
>>> 5101520
>>> print(a, b, a + b, end="20\n", sep="")
>>> 5101520
```

Command Parameters

As long as you mark `sep` and `end` clearly **and** after the things you want to print, the ordering doesn't matter!