## Steps to install Python on `windows` machine

- Go to official website of Python
- Click on “downloads” tab and then select “windows”
- Click on Latest Python 3 Release - Python 3.10.7
- Scroll down to “Files” section
    - Last 2 files

  Windows installer (32-bit)

  Windows installer (64-bit)-

You can decide which installer to pick based on which “system type” you are using.

Right click on “This PC (my computer)” and check properties to see if it is 32 bit or 64 bit.

*NOTE*
Select checkbox called “Add Python to PATH” while installing Python using this installer.

LAST STEPs to verify “Python installation”

- Go to “start” button, search for “CMD”
- Open application called “command prompt”
- Type “python” and hit enter button. If you see Python interpreter has opened then your installation is successful.

## General Concept of Variables

- Variable is kind of a label
- Variable is something that can change
- Variable is way of referring to a memory location. This memory location contains values.
- Variable is just a reference
- You can choose variable name except for words from keyword (there are other rules like pep8 standards, we will learn about them later)

- You can think of variable as a “container” to contain some kind of value
- Value of variable may change during program execution.
- Also, type() of variable can change during program execution
  Example ::

```
i = 100          # variable `i` is automatically set to an integer (concept of
i = i + 0.11       # value of variable `i` has changed in this line
i = str(i)     # and now variable `i` will be "type casted" into string
```

## Variables in Python

- We do NOT declare variables in Python

## Keywords

- Keywords are words reserved by the programming language.
- We can *NOT* use keywords as variables as “identifiers”.

NOTE - “Identifiers” are variable names, function names, class names,... etc.

```
help("keywords")
```

```
False      class      finally    is         return
None       continue   for        lambda     try
True       def        from       nonlocal   while
and        del        global     not        with
as         elif       if         or         yield
assert     else       import     pass
break      except     in         raise
```

## Type casting

Changing type of object from one type to another.

float(string) -> float value
int(string) -> integer value

str(integer) or str(float) -> string representation

Example -

```
int("12")
float("20.5")
str(12)
bool("prashant")
```

Rules for type casting into Boolean object -

- non-zero integer is treated as `True`
- non-empty string is treated as `True`
- non-empty list is tread as `True`
- In short, any non-empty “collection” is treated as `True`

## Reading input from keyboard

Real life software / application interacts with user for data input.
In Python, we use in-built function named `input()` .
- `input()` function returns a string as output. In another words, it returns a object of class `str` .

Let’s create a `application` (OR call it `program` to accept 2 numbers from user and perform addition of them.

```
first_number = input("Please enter first number : ")
second_number = input("Please enter second number : ")

# type cast `str` value into `int` value so that addition can be performed usi
first_number = int(first_number)
second_number = int(second_number)

# addition using + operator
result = first_number + second_number
print(result)
```

## Multiple assignments in a single line

```
number1, number2 = 10, 20
print(number1)
print(number2)
```

## Exercise

1. Write a program that will convert temperature to Celsius from Fahrenheit by using formula given below

```
C=(F-32)/1.8

# HINT
# step 1: Take input from user. Inform user to enter value in Fahrenheit. Inpu
# step 2 : Type cast value entered into `float`
# step 3: Apply formula using mathematical operators
```

2. Swap 2 numbers using multiple assignment (apply concept of tuple packing and unpacking to swap 2 numbers.

```
# python code
a = 1000
b = 20

# HINT
# Perform swap in a single line code
# Perform swap such that a will have value of b AND b will have value of a
# meaning, a should be 20 and b should become 1000
```

3. What built in functions have we learned so far?

```
write your answer here
```

4. What are “survival functions”? Is "survival function a standard name OR name given by us?

```
write your answer here
```

5. When we invoke type() function on any object, how do we describe the result?

write your answer here

```
# HINT (python code)
>>> foo = 10
>>> type(foo)
int

# We describe this like following statement
# variable `foo` is object of class `int`

# PERFORM THIS FOR OTHER DATA TYPES
```

6. What is use of `dir()` function?

write your answer here

```
# HINT (python code)
>>> foo = ""   # empty string
>>> dir(foo)
['__add__',
 '__class__',
 '__contains__',
 ...
 ...
 ...
 'capitalize',
 'casefold',
 'split',
 'join',
 'lower',
 'upper',
 'find',
 'count',
 'lstrip',
 'rstrip',
 'endswith',
 'startswith',
 'index',
 'isalnum',
 ...
```

```
 ...
 ]

# We invoke `help()` function on all of these functions of class `str`
# We call `help()` function by passing class attribute as input
# For example -

# PYTHON CODE
foo = ""   # empty string
help(foo.startswith)
```

Output -

```
Help on built-in function startswith:

startswith(...) method of builtins.str instance
    S.startswith(prefix[, start[, end]]) -> bool

    Return True if S starts with the specified prefix, False otherwise.
    With optional start, test S beginning at that position.
    With optional end, stop comparing S at that position.
    prefix can also be a tuple of strings to try.
```

Read output carefully and practice each function of `string` by taking various string examples yourself.
If you don't understand, please ask question in the session.

7. What is indexing? What data types can we can apply concept of indexing on?

```
write your answer here
```

8. Use slicing concept to slice out my name and concatenate your name in the following string

```
# python code
greetings = "Good morning, Prashant"

# You should change value of variable `greetings` such that output becomes fol
# "Good morning, <your-own-name-here>"
```

9. Take some examples of palindrome and use string reversal technique `[::-]` AND equality operator `==` to prove it's palindrome.
   Example -

```
# python code
example1 = "mom"
reverse_ = example1[::-1]

example1 == reverse_   # output should be `True`
```