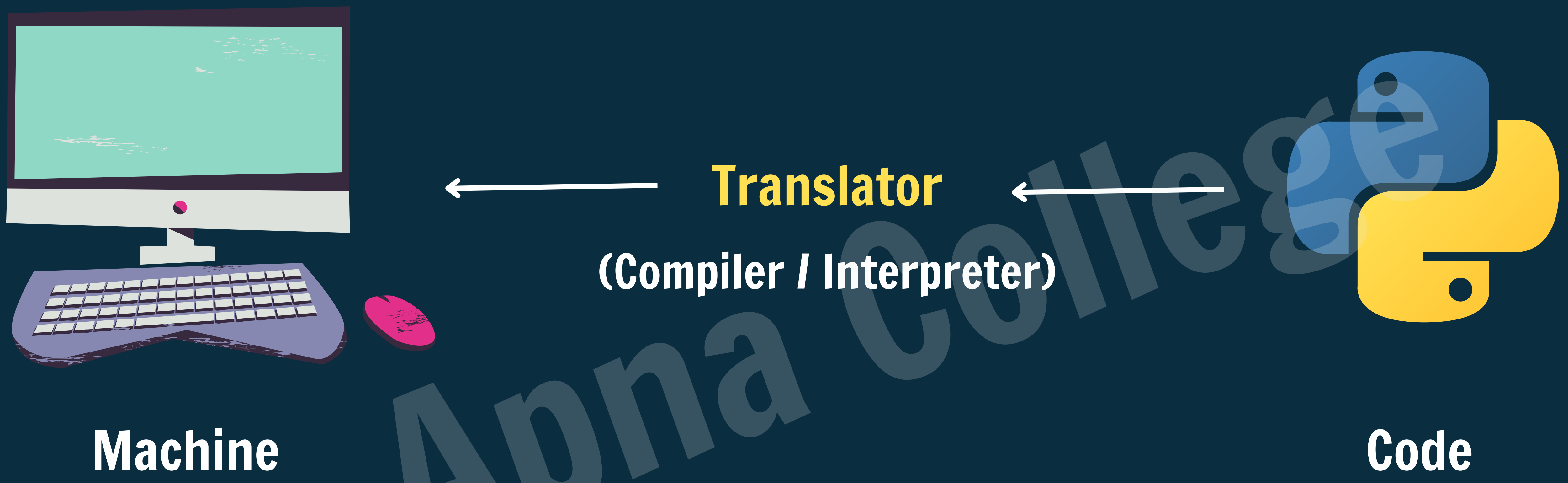
Programming



What is Python?

- Python is simple & easy
- Free & Open Source
- High Level Language
- Developed by Guido van Rossum
- Portable

Our First Program

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

Python Character Set

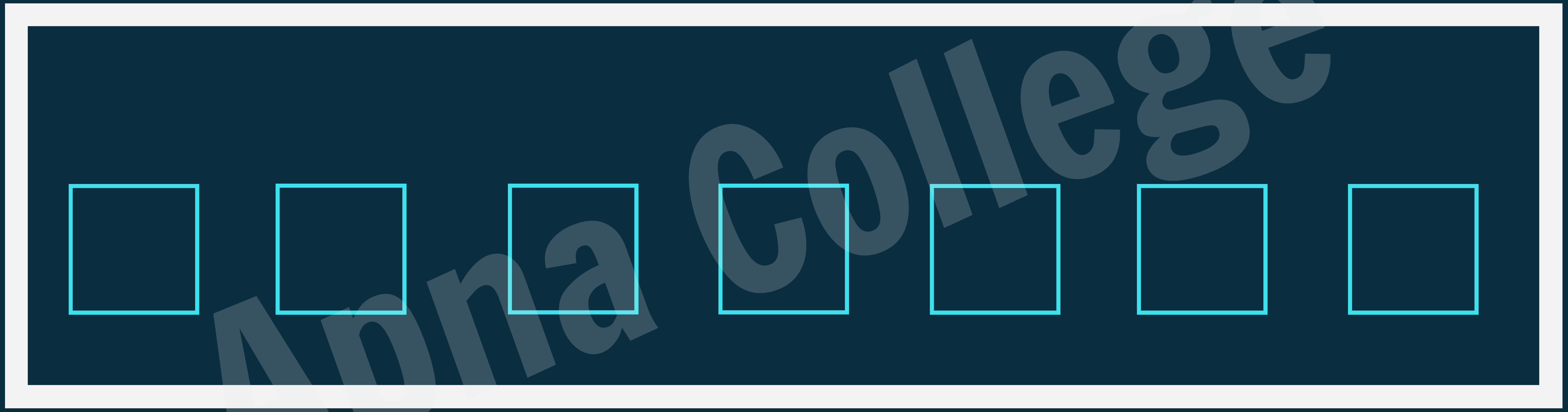
- Letters - A to Z, a to z
- Digits - 0 to 9
- Special Symbols - + - * / etc.
- Whitespaces - Blank Space, tab, carriage return, newline, formfeed
- Other characters - Python can process all ASCII and Unicode characters as part of data or literals

Variables

A variable is a name given to a memory location in a program.

```
name = "Shradha"
age = 23
price = 25.99
```

Memory



```
name = "Shradha"

age = 23

price = 25.99
```

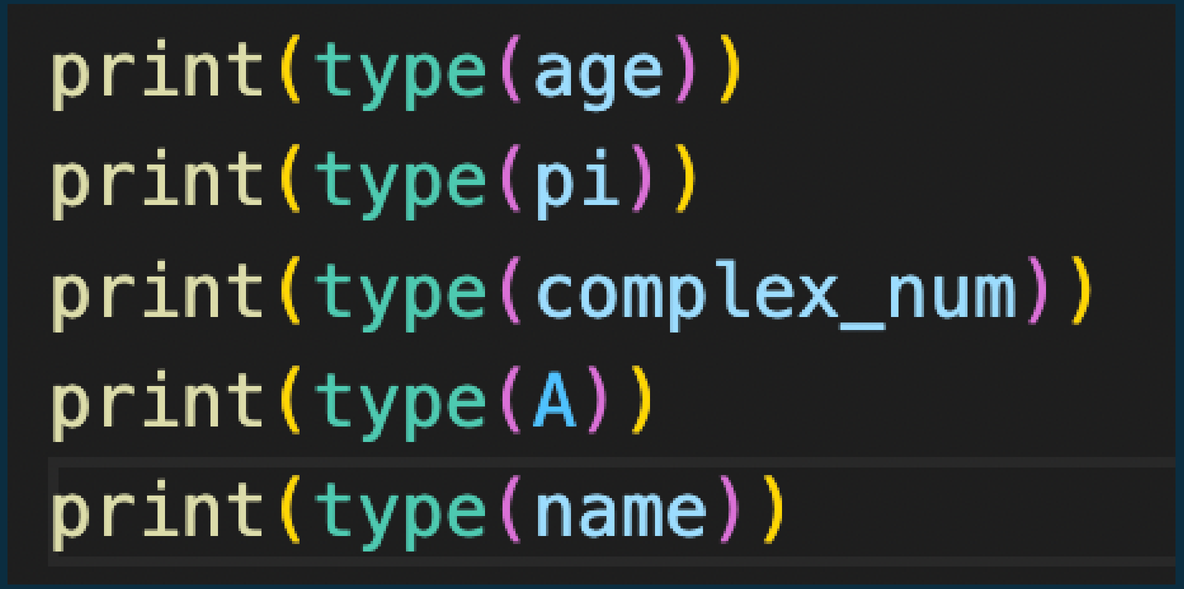
Rules for Identifiers

1. Identifiers can be combination of uppercase and lowercase letters, digits or an underscore(_). So **myVariable**, **variable_1**, **variable_for_print** all are valid python identifiers.
2. An Identifier can not start with digit. So while **variable1** is valid, **1variable** is not valid.
3. We can't use special symbols like !,#,@,%,$ etc in our Identifier.
4. Identifier can be of any length.

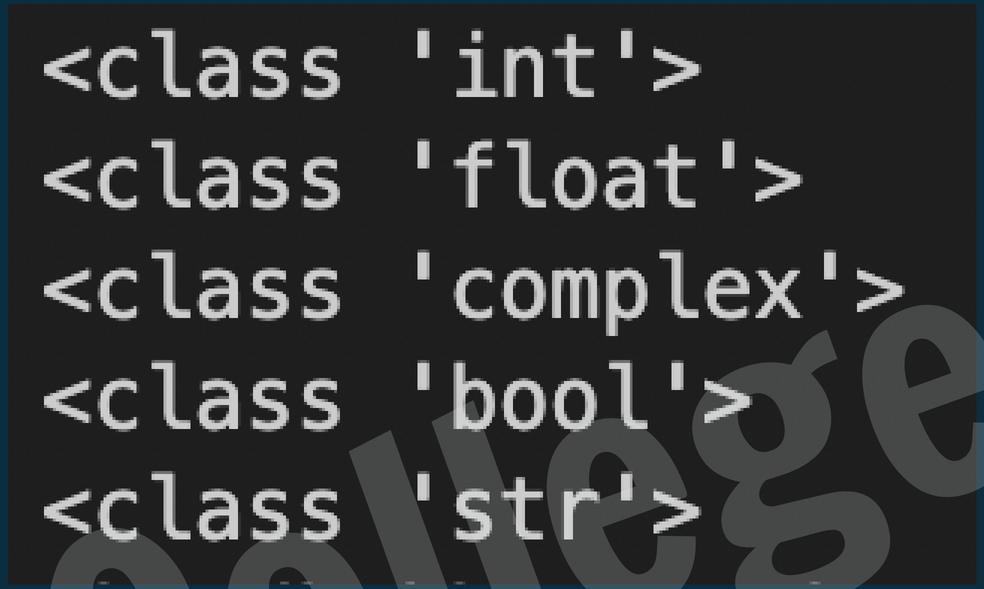
Data Types

- Integers
- String
- Float
- Boolean
- None

Data Types



```python
print(type(age))
print(type(pi))
print(type(complex_num))
print(type(A))
print(type(name))
```



```
<class 'int'>
<class 'float'>
<class 'complex'>
<class 'bool'>
<class 'str'>
```

Keywords

Keywords are reserved words in python.

*False should be uppercase

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

Print Sum

Comments in Python

```
# Single Line Comment


"""
Multi Line
Comment
"""
```

Types of Operators

An operator is a symbol that performs a certain operation between operands.

- Arithmetic Operators (+ , - , * , / , % , **)
- Relational / Comparison Operators (== , != , > , < , >= , <=)
- Assignment Operators (= , +=, -= , *= , /= , %= , **=)
- Logical Operators (not , and , or)

Type Conversion

```
a, b = 1, 2.0
sum = a + b



#error
a, b = 1, "2"
sum = a + b
```

Type Casting

```
a, b = 1, "2"
c = int(b)
sum = a + c
```

Type Casting

Function	Description
int(y [base])	It converts *y* to an integer, and Base specifies the number base. For example, if you want to convert the string in decimal numbers then you'll use 10 as base.
float(y)	It converts *y* to a floating-point number.
complex(real [imag])	It creates a complex number.
str(y)	It converts *y* to a string.
tuple(y)	It converts *y* to a tuple.
list(y)	It converts *y* to a list.
set(y)	It converts *y* to a set.
dict(y)	It creates a dictionary and *y* should be a sequence of (key, value) tuples.
ord(y)	It converts a character into an integer.
hex(y)	It converts an integer to a hexadecimal string.
oct(y)	It converts an integer to an octal string

Input in Python

input( ) statement is used to accept values (using keyboard) from user

```
input( )  #result for input( ) is always a str

int ( input( ) )  #int

float ( input( ) )   #float
```

give code eg of all 3

Let‘s Practice

Write a Program to input 2 numbers & print their sum.

Let‘s Practice

WAP to input side of a square & print its area.

Let‘s Practice

WAP to input 2 floating point numbers & print their average.

Let‘s Practice

WAP to input 2 int numbers, a and b.

Print True if a is greater than or equal to b. If not print False.

Strings

String is data type that stores a sequence of characters.

Basic Operations

- **concatenation**

  "hello" + "world" $\longrightarrow$ "helloworld"

- **length of str**

  len(str)

Indexing

A	p	n	a	_	C	o	l	l	e	g	e
0	1	2	3	4	5	6	7	8	9	10	11

str = “Apna_College”

str[0] is ‘A’, str[1] is ‘p’ ...

str[0] = ‘B’ #not allowed

Slicing

Accessing parts of a string

str[starting_idx : ending_idx] #ending idx is not included

str = "ApnaCollege"

str[1 : 4] is "pna"

str[: 4] is same as str[0 : 4]

str[1 :] is same as str[1 : len(str)]

Slicing

Negative Index

A	p	p	l	e
-5	-4	-3	-2	-1

str = “Apple”

str[-3 : -1] is “pl”

String Functions

```
str = “I am a coder.”

str.endsWith(“er.“)  #returns true if string ends with substr

str.capitalize( )  #capitalizes 1st char

str.replace( old, new )  #replaces all occurrences of old with new

str.find( word )  #returns 1st index of 1st occurrence

str.count(“am“)  #counts the occurrence of substr in string
```

Let‘s Practice

WAP to input user’s first name & print its length.

WAP to find the occurrence of ‘$’ in a String.

Conditional Statements

if-elif-else (SYNTAX)

```
if(condition) :
        Statement1
elif(condition):
        Statement2
else:
        StatementN
```

Conditional Statements

Grade students based on marks

marks >= 90, grade = “A”

90 > marks >= 80, grade = “B”

80 > marks >= 70, grade = “C”

70 > marks, grade = “D”

Let's Practice

WAP to check if a number entered by the user is odd or even.

WAP to find the greatest of 3 numbers entered by the user.

WAP to check if a number is a multiple of 7 or not.