Basics of Programming in Python and CPP

URVASHI BHARGAVA

Y TERESHA

Where Python is used

Basic Syntax

Printing to the Screen:

```
print ("Hello, Python!")
```

Reading Keyboard Input:

```
name = input("Enter your name: ")
```

Comments

- Single line:

```
# This is a comment.
```

- Multiple lines:

```
'''
print("We are in a comment")
print ("We are still in a comment")
'''
```

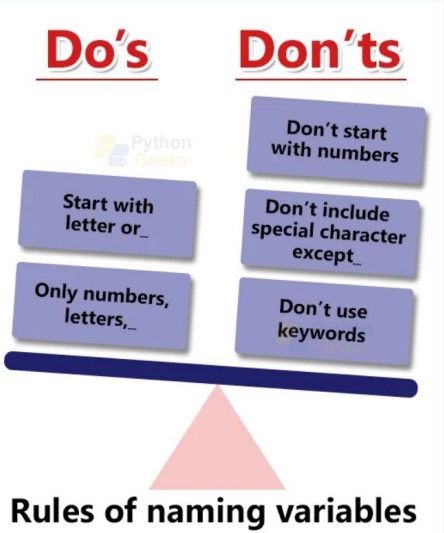
Variables

Python is dynamically typed. You do not need to declare variables!

```
counter = 100          # An integer assignment
miles   = 1000.0       # A floating point
name    = "John"       # A string
z = None               # A null value
```

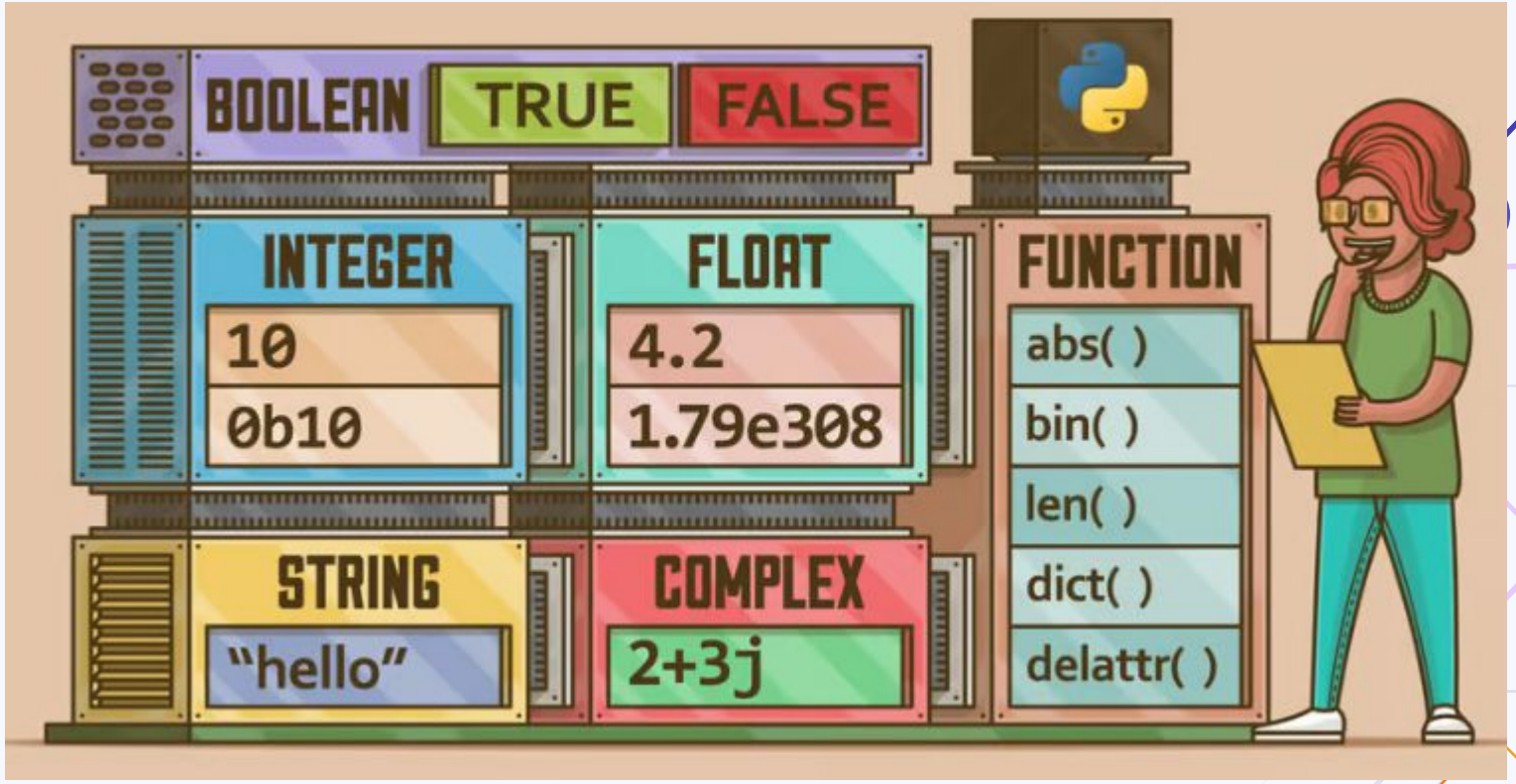
keywords in Python

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



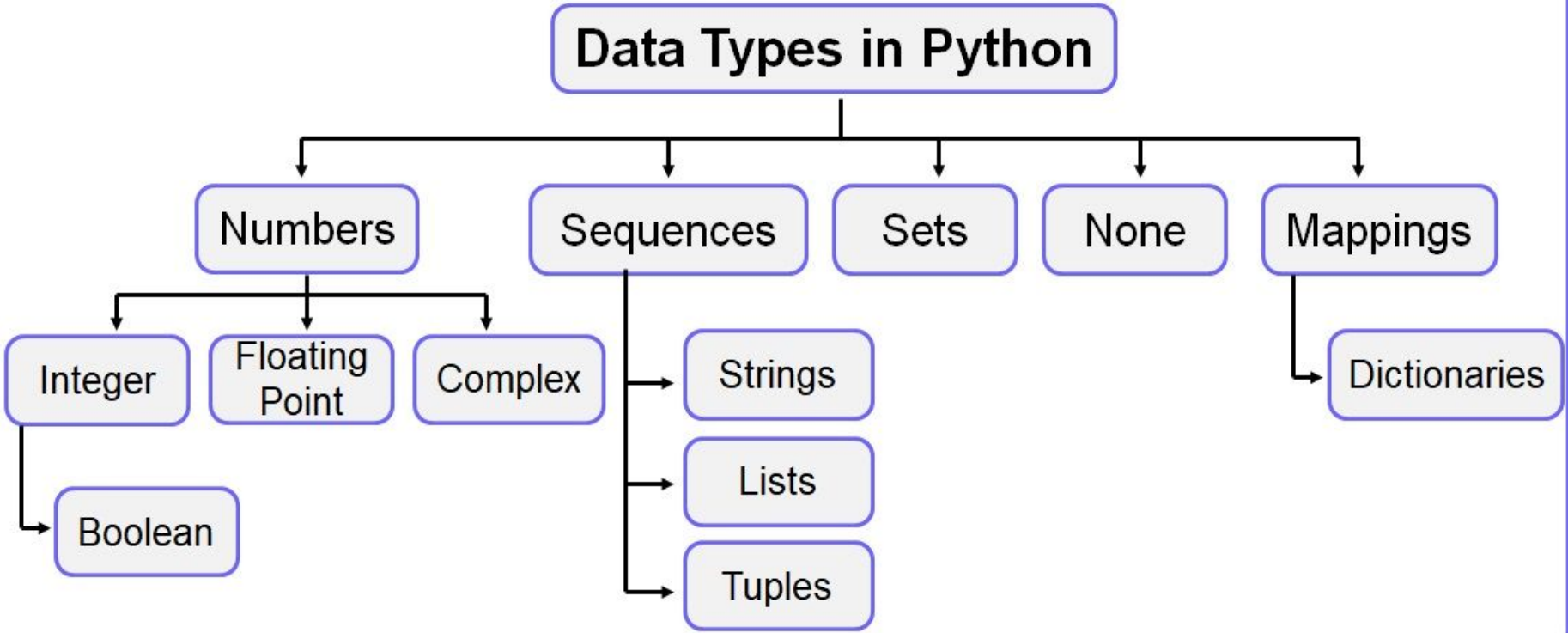
Rules of naming variables

Data Types


BOOLEAN
TRUE
FALSE
INTEGER
10
0b10
FLOAT
4.2
1.79e308
FUNCTION
abs( )
bin( )
len( )
dict( )
delattr( )
STRING
"hello"
COMPLEX
2+3j

teachoo

Data Types in Python



Data Types

1) Lists
   - Lists are used to store multiple items in a single variable
     `mylist = ["apple", "banana", "cherry"]`
   - List items are ordered, changeable, and allow duplicate values.
   - List items can be of any data type,and can contain different data types
   - It is also possible to use the `list()` constructor when creating a new list.

```
thislist = list(("apple", "banana", "cherry"))
```

2) Tuples
   - Tuple items are ordered, unchangeable, and allow duplicate values.
   - To create a tuple with only one item, you have to add a comma after the item.
     `thistuple = ("apple",)`
     `print(type(thistuple))OUTPUT: <class 'tuple'>`
   - Tuple items can be of any data type and can contain different data types
   - It is also possible to use the `tuple()` constructor to make a tuple.
     `thistuple = tuple(("apple", "banana", "cherry"))`

Data Types

3) Dictionary

- Dictionaries are used to store data values in key:value pairs and can be referred to by using the key name.
- A dictionary is a collection which is ordered, changeable and do not allow duplicates.

```
thisdict = {
  "brand": "Ford",
  "electric": False,
  "year": 1964,
  "colors": ["red", "white", "blue"]
}
```

- It is also possible to use the `dict()` constructor to make a dictionary.

```
thisdict = dict(name = "John", age = 36,
country = "Norway")
```

4) Sets

- A set is a collection which is *unordered*, *unchangeable**, and cannot have duplicates
- Sets are written with curly brackets.

```
thisset = {"apple", "banana", "cherry"}
```

- Set items can be of any data type and can contain different data types
- It is also possible to use the `set()` constructor to make a set.

```
thisset = set(("apple", "banana", "cherry"))
```

Data Types

Example	Data Type
`x = "Hello World"`	str
`x = 20`	int
`x = 20.5`	float
`x = 1j`	complex
`x = ["apple", "banana", "cherry"]`	list
`x = ("apple", "banana", "cherry")`	tuple
`x = range(6)`	range
`x = {"name" : "John", "age" : 36}`	dict
`x = {"apple", "banana", "cherry"}`	set

String Manipulation

1) + operator - concatenation

```
>>> print('Go team' + '!!!')
Go team!!!
```

2) * operator - for creating multiple copies

```
>>> s = 'foo.'

>>> s * 4
'foo.foo.foo.foo.'
```

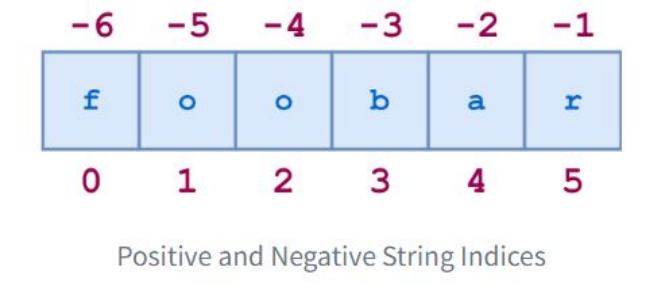
3) in operator - membership operator. True if exists, else false

```
>>> s = 'foo'

>>> s in 'That\'s food for thought.'
True
>>> s in 'That\'s good for now.'
False
```

Function	Description
`chr()`	Converts an integer to a character
`ord()`	Converts a character to an integer
`len()`	Returns the length of a string
`str()`	Returns a string representation of an object

```
x = chr(97)

x = ord("h")
```

String Manipulation - Slicing



Positive and Negative String Indices

```
>>> s = 'foobar'
>>> s[2:5]
'oba'
```

```
>>> s[:4]
'foob'
>>> s[0:4]
'foob'
```

```
>>> s[2:]
'obar'
>>> s[2:len(s)]
'obar'
```

```
>>> s[0:6:2]
'foa'

>>> s[1:6:2]
'obr'
```

```
>>> s[-5:-2] == s[1:4]
True
```

Conditional Statements- if elif else

if Statement

```
if expression:
    statement(s)
```

if..else Statement

```
if expression:
    statement(s)
else:
    statement(s)
```

if..elif..else Statement

```
if expression1:
    statement(s)
elif expression2:
    statement(s)
elif expression3:
    statement(s)
else:
    statement(s)
```

```
a, b = 4, 5

if a < b:
    x = 'smaller'
else:
    x = 'bigger'

print (x)
```

→

```
x = 'smaller' if a < b else 'bigger'
```

Conditional Statements- for and while

```
# First Example
for letter in 'Python':
   print ('Current Letter :', letter)
```

```
Current Letter : P
Current Letter : y
Current Letter : t
Current Letter : h
Current Letter : o
Current Letter : n
```

```
food = ['pizza', 'steak'.  'rice']
for index in range(len( food )):     # range(3) iterates between 0 to 2
   print ('Current food :', food[index])
```

```
Current food : pizza
Current food : steak
Current food : rice
```

```
count = 0
while (count < 5):
   print ('The count is:', count)
   count = count + 1
```

```
The count is: 0
The count is: 1
The count is: 2
The count is: 3
The count is: 4
```

Loop Control Statements

- **break** :Terminates the loop statement and transfers execution to the statement immediately following the loop.

```
for letter in 'Python':
   if letter == 'h':
      break
   print ('Current Letter :', letter)
```

```
Current Letter : P
Current Letter : y
Current Letter : t
```

- **continue** :Causes the loop to skip the remainder of its body and immediately retest its condition prior to reiterating.

```
for letter in 'Python':
   if letter == 'h':
      continue
   print ('Current Letter :', letter)
```

```
Current Letter : P
Current Letter : y
Current Letter : t
Current Letter : o
Current Letter : n
```

Functions

QUESTION:
CREATE A FUNCTION TO PRINT YOUR NAME(FIRST NAME AND LAST NAME)

Function Syntax

```
def functionname( parameters ):
   "function_docstring"
   function_statements
   return [expression]
```

```
def printme( str ):
   "This prints a passed string into this function"
   print str
   return
```

Function Arguments

You can call a function by using any of the following types of arguments:

- **Required arguments**: the arguments passed to the function in correct positional order.

```
def func( name, age ):
....
func("Alex", 50)
```

- **Keyword arguments**: the function call identifies the arguments by the parameter names.

```
def func( name, age ):
....
func( age=50, name="Alex" )
```

- **Default arguments**: the argument has a default value in the function declaration used when the value is not provided in the function call.

```
def func( name, age = 35 ):
...
func( "Alex" )
```

Modules

```
import <module_name>
```

From math import sqrt
print(sqrt(25))

Output:
5

```
import math
print(math.pi)
print(math.sin(math.pi / 2))
```

In-built modules:

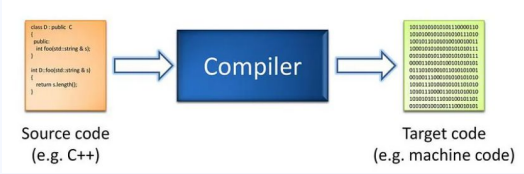
- Math: useful to perform complex mathematical functions
- Os: Provides a way to interact with the underlying operating system, such as reading or writing files, executing shell commands, and working with directories.
- Date: used to work with date and time
- Sys: access to some variables used or maintained by the Python interpreter, such as the command-line arguments passed to the script, the Python version, and the location of the Python executable.

User-Defined modules:

```
def add(a, b):
    return a + b

def sub(a, b):
    return a - b

def mul(a, b):
    return a * b

def div(a, b):
    return a / b
```

```
import calculator

print("Addition of 5 and 4 is:", calculator.add(5, 4))
print("Subtraction of 7 and 2 is:", calculator.sub(7, 2))
print("Multiplication of 3 and 4 is:", calculator.mul(3, 4))
print("Division of 12 and 3 is:", calculator.div(12, 3))
```

BASICS OF C++



Where C++ is used

Fortnite? Call of Duty? Clash of Clans?

Basic Syntax

Q) Take the name of the user as input and display it

Input: cin >>

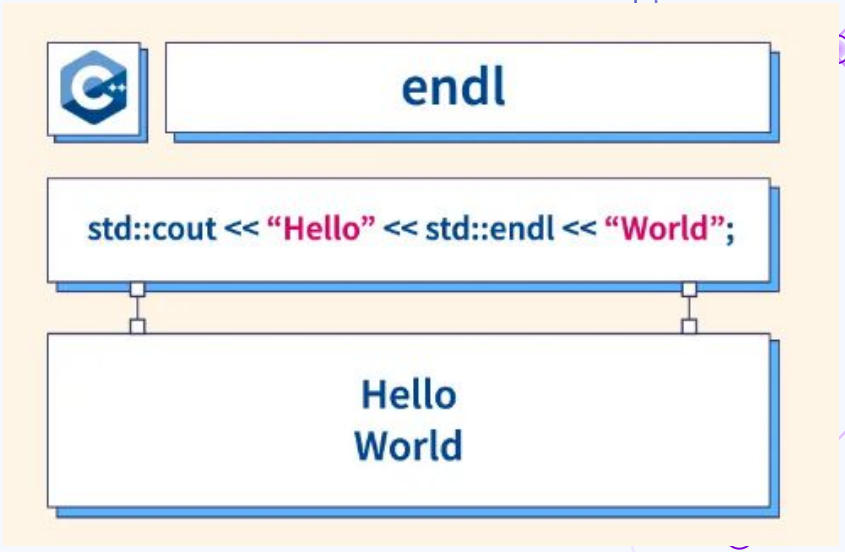
Output: cout <<

	1	`#include <iostream>`	Header File
	2	`using namespace std;`	Standard Namespace
	3	`int main()`	Main Function
FUNCTION BODY	4	`{`	
	5	`int num1 = 24;` `int num2 = 34;`	Declaration of Variable
	6	`int result = num1 + num2;`	Expressions
	7	`cout << result << endl;`	Output
	8	`return 0;`	Return Statement
	9	`}`	

Comments

// Single line comment

/* Multi-line comment */



Data Types

Type	Usage	Examples
int	integer numbers	0 420
double	floating-point numbers	3.14 -200.0
char	characters	'a' '@'
string	sequence of characters	"Hello World!" "Codecademy"
bool	truth values	true false

Numbers with decimal points

Floating-Point Types

Type	Storage size	Value range
float	4 byte	1.2E-38 to 3.4E+38
double	8 byte	2.3E-308 to 1.7E+308
long double	10 byte	3.4E-4932 to 1.1E+4932

‘ ‘ for character

“ “ for string

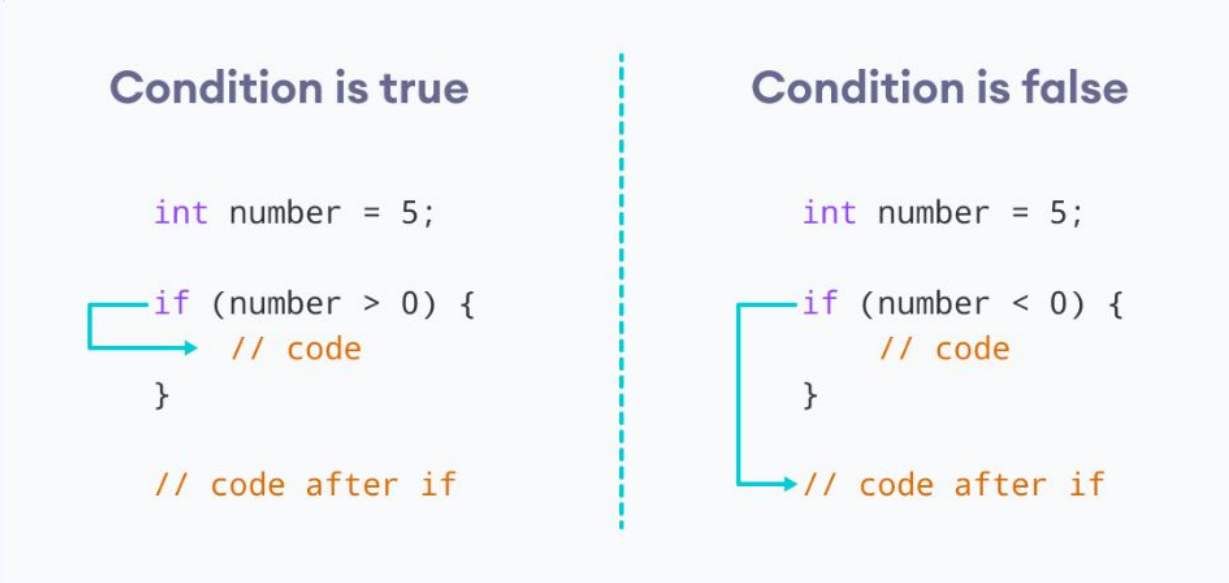
Think about it...

float y = 5;
cout<<y;

Will it print 5? 5.0? or we can’t store an int in a float?

If-Else

Q) Write a program that takes input as age (int) and tells if the user is an adult or not



How if Statement Works

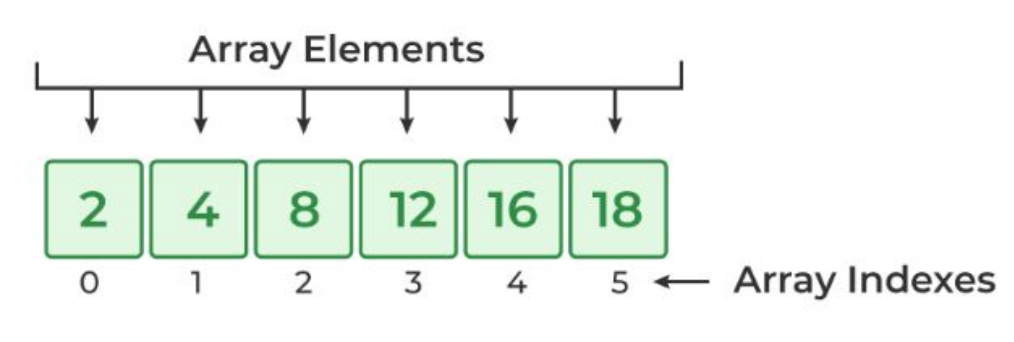
18+ ADULTS ONLY

Switch

Q) Given a number, print the day of the week

```
switch(a + b)
{
    case 2:
        // executes if a + b = 2

    case 4:
        // executes if a + b = 4

    case 6:
        // executes if a + b = 6

    default:
        // executes if a + b doesn't equal 2, 4, or 6 (doesn't fit any
other cases)
}
```

Arrays - similar data types together




```
int arr[6];

Can modify values too!
arr[0]+=3;
cout<<arr[0];

int arr[3][5];
```

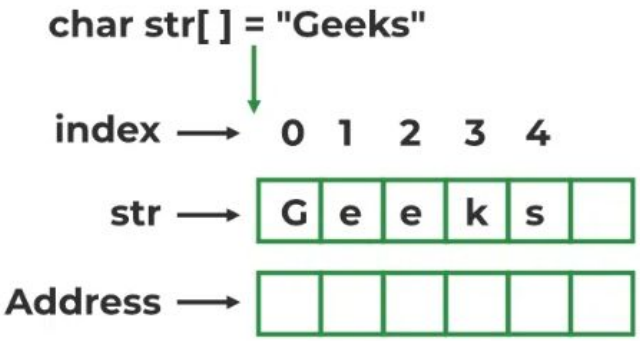
	column 1	column 2	column 3	column 4	column 5
row1	arr[0][0]	arr[0][1]	arr[0][2]	arr[0][3]	arr[0][4]
row2	arr[1][0]	arr[1][1]	arr[1][2]	arr[1][3]	arr[1][4]
row3	arr[2][0]	arr[2][1]	arr[2][2]	arr[2][3]	arr[2][4]

Beginnersbook.com

Strings

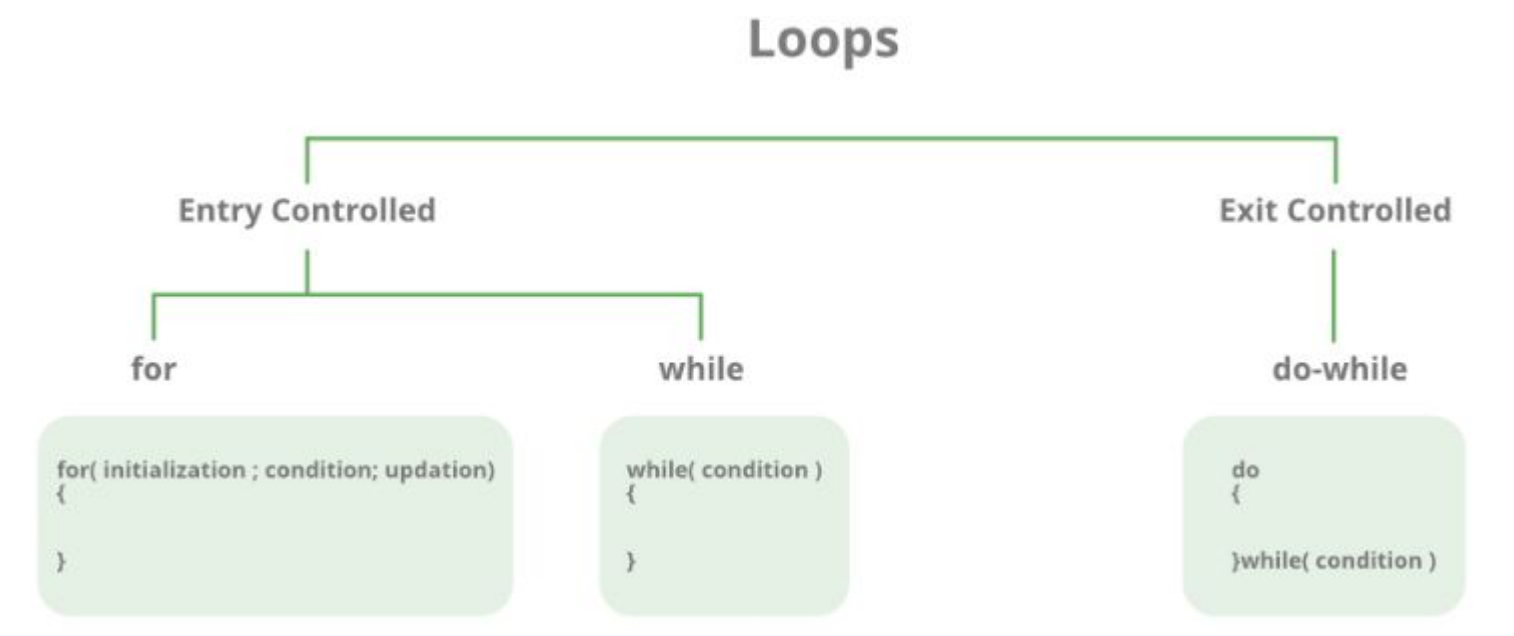
```
string str = ("gfg");
string str = "gfg";
string str; str = "gfg";
```

```
string s;
cin>>s;
cout<<s;
getline(cin,s);
cout<<s;
```

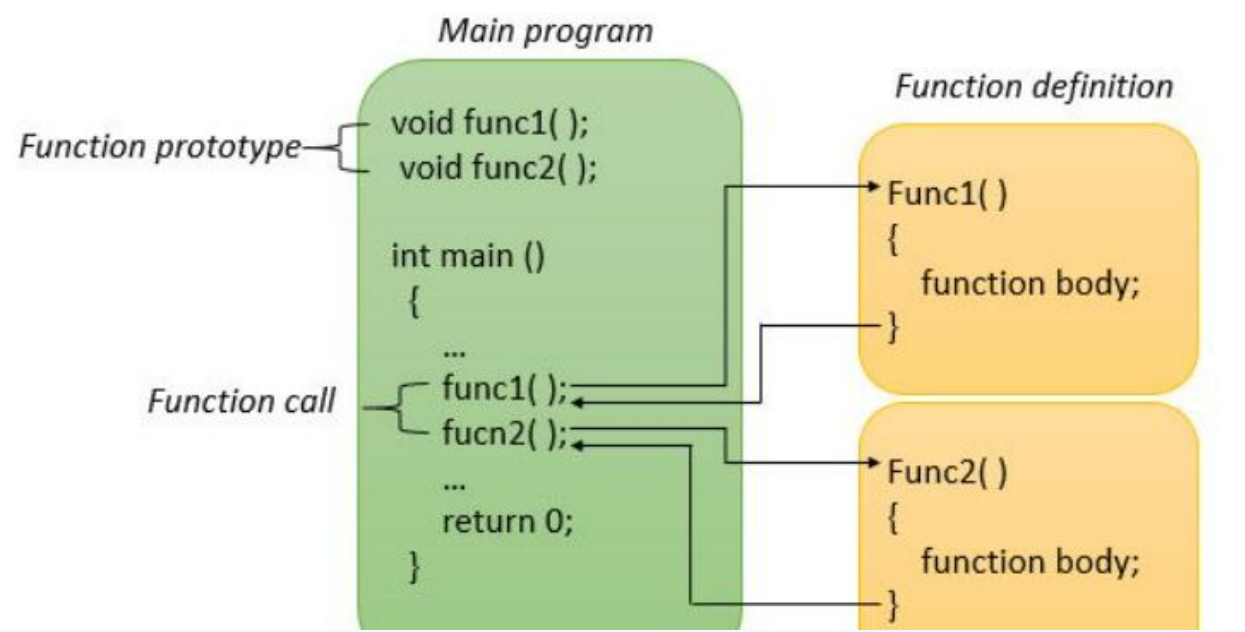


Loops

Q) Print “Welcome to this workshop” 50 times.



Functions



Pass by value / Pass by reference

Project Ideas in Python - Beginner

1) HANGMAN

Concepts used: Dictionaries, lists,modules, user input,string manipulation, conditional statements, loops, functions

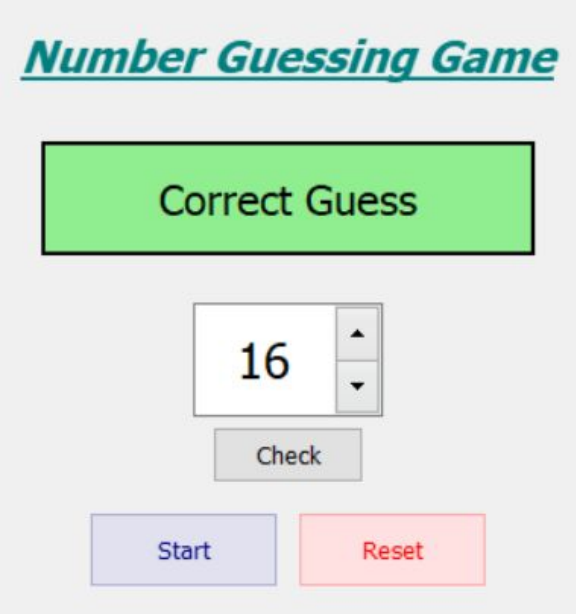
2) Rock Paper Scissors

Concepts used: user input, randomization, conditional statements (if-elif-else), loops (optional for multiple rounds), functions

Project Ideas in C++ - Beginner

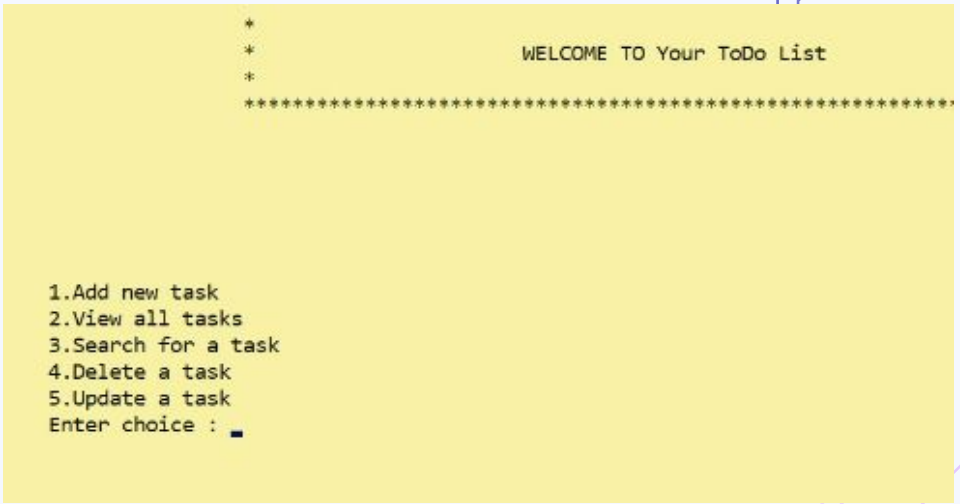
1) Guess the Number Game

Concepts used: Random number generation, user input/output, conditional statements.



2) Simple To - Do List

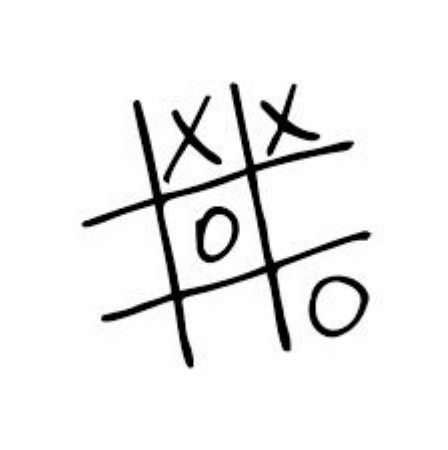
Concepts used: Arrays or linked lists, user input/output, basic file I/O.



Project Ideas in Python - Intermediate

3) Tic-Tac-Toe

Concepts used: Conditional Statements,modules,Gui-Tkinter



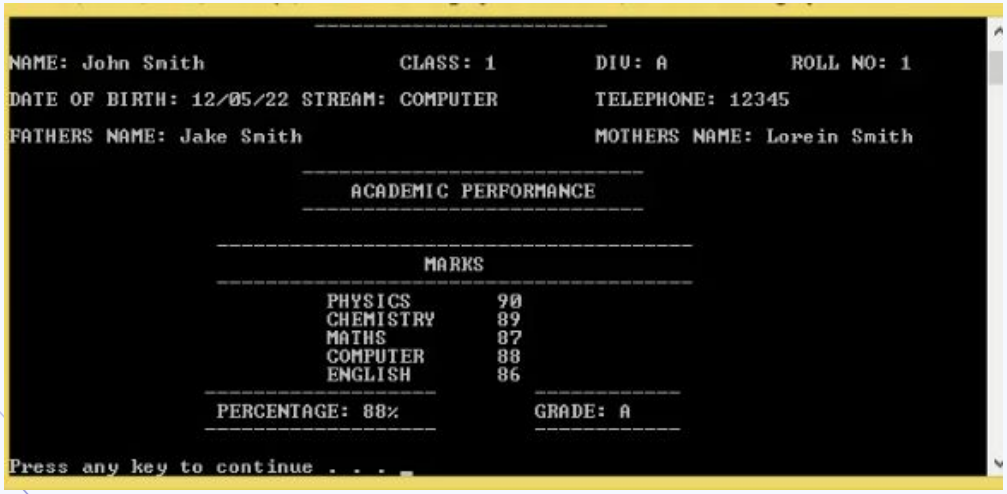
4) QR Code encoder/decoder

Concepts used: modules

Project Ideas in C++ - Intermediate

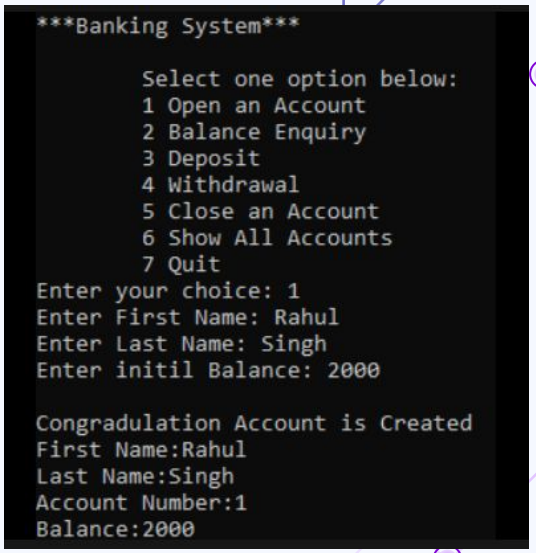
3) Student Grade Tracker

Concepts used: Arrays or data structures (arrays of structures), user input/output, mathematical calculations.



4) Banking System

Concepts used: Object-oriented programming (classes, encapsulation), file handling (for data persistence), user input validation, transaction processing



Project Ideas in Python - Advanced

5) Whatsapp Birthday bot

Concepts used: web scraping, JSON manipulation, and automation.

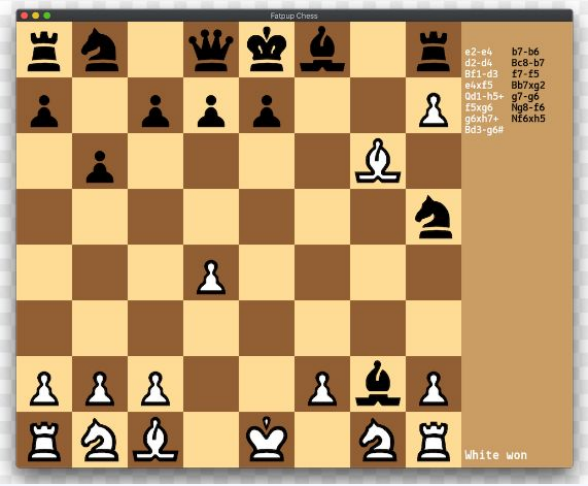
6) Alarm Clock

Concepts used: modules(time handling), use input, loops, conditional statements, and potentially using external libraries for sound graphical interfaces.

Project Ideas in C++ - Advanced

5) Chess Game

Concepts used: Object-oriented programming (classes, inheritance, polymorphism), 2D arrays (for representing the chessboard), loops, conditional statements (for game logic), user input/output



6) Text-Based Role-Playing Game

Concepts used: Object-oriented programming (classes, inheritance, polymorphism), file I/O, game logic.

```
william@mysixtyfour:~/Programming/commandlineprogs/txtadv_1$ ./a.out
What shall I do? look
Your raw command was look
LOOK

What shall I do? go north
Your raw command was go north
GO NORTH

What shall I do? west
Your raw command was west
WEST

What shall I do? eat some lunch and take a nap
Your raw command was eat some lunch and take a nap
Command too long. Only type one or two words (direction or verb and noun)

What shall I do? get key
Your raw command was get key
GET KEY
```

Practical exercise - calculator

Objective:

Introduce participants to basic coding concepts by building a **simple calculator**

Practical exercise - calculator In python

Objective:

Introduce participants to basic coding concepts by building a **simple calculator.**

```
# Simple Calculator
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
print("Sum:", num1 + num2)
print("Difference:", num1 - num2)
print("Product:", num1 * num2)
print("Quotient:", num1 / num2)
```

Practical exercise - calculator In C++

Objective:

Introduce participants to basic coding concepts by building a **simple calculator.**

```
// Simple Calculator
int num1, num2;
cout<<"Enter the first number: ";
cin>>num1;
cout<<"Enter the second number: ";
cin>>num2;
cout<<"Sum: "<<(num1 + num2);
cout<<"Difference: "<<(num1 - num2);
cout<<"Product: "<<(num1 * num2);
cout<<"Quotient: "<<(num1 / num2);
```

THANK YOU

Feel free to contact us anytime :)

Urvashi Bhargava

EMAIL:
urvashi122003@gmail.com

GITHUB:
https://github.com/urvashii-b

LINKEDIN:
https://www.linkedin.com/in/urvashi-bhargava-b66100262/

Y Teresha

EMAIL:
teresha1903@gmail.com

GITHUB:
https://github.com/Tereshaa

LINKEDIN:
https://www.linkedin.com/in/y-teresha1903/