ARTIFICIAL INTELIGENCE

Name: M. Byula

Pino: HU21CSEN0500190

Experiment-1

Date:18-07-23

1. **Revisit/Refresh the study of Python.**

BASICS OF PYTHON

1. **Syntax:** Python uses indentation to define blocks of code, instead of braces {}. It enhances readability.
2. **Variables:** Variables are containers that store data values.
3. **Data Types:** Common data types include int, float, str, Boolean, and more.
4. **Print:** Display output using "print ()."
5. **Comments:** Use # for single the comments, & '', "", for multi-line comments.
6. **Arithmetic operators:** addition (+), subtraction (-), multiplication (*), and division (/). For integer device //, "%" for modulo, "**" for exponential.
7. **Comparison operators:** "==," "! =", "<", ">"," < =",">=" to compare values.
8. **Logical operators**: 'and', 'or', 'not' for logical operations.
9. **Conditional statements**: use 'if', 'elif', and 'else' to perform conditional actions.
10. **Loops**: i. 'for' loop: Iterates over elements in a sequence
    ii. 'while' loop: Repeats code while a condition is true.
11. **Lists**: Ordered mutable collection of elements enclosed in '[]'.
    L = [1, 2, 3]
12. **Dictionaries**: Unordered key-value pairs enclosed in '{}'.
    Ex: person= {"name": "Ram", "class": "CSE"}
13. **Tuples**: Like lists but immutable enclosed in '()'.
    Ex: Coordinates = (1,2)
14. **Sets**: A set is a collection that is unordered, unchanged, and unindexed and stores multiple items in a single variable.
    S = {"apple", "banana"}.
15. **Functions:** Define reusable code blocks using 'def'.
16. **Modules:** Organize code in separate files, and import them using 'import'.
17. **Exceptions:** Handle errors using 'try', and 'except'.
18. **File I/O:** Read from and write to files using 'open ()'.
19. **Classes and objects:** Create custom data types using classes & and instantiate objects.
20. **Inheritance:** Classes can be inherit attributes and methods from other classes.

21. **Packages:** Collections of related modules installed using packages manages like 'pip'.

List: -

- Lists are used to store multiple items in a single variable.
- Lists are one of 4 built-in data types in Python used to store collections of data.
- Lists are created using square brackets.
- List items are
    1. Ordered: The items in the list have a defined order; that order will not change. If new items to a list are added, the new items will be placed at the end of the list.
    2. Changeable: We can change, add, and remove items in a list after it has been created.
    3. Allow duplicates: Since lists are indexed, lists can have items with the same value.

Methods in the list: -

append (), clear (), copy (), count (), extend (), index () insert (), pop (), remove (), reverse (), sort ()

Tuple: -

- Tuple items are ordered, unchangeable, and allow duplicate values.
- Tuple items are
    1. Ordered: All the items in a tuple have a definite order, and that order will not change.
    2. Unchangeable: Tuples are unchangeable, meaning that we cannot change, add, or remove items after the tuple has been created.
    3. Allow Duplicates: Since tuples are indexed, they can have items with the same value.

Methods in tuple:

count (), index (), are two built-in methods in tuples.

Sets: - set items are unordered, unchangeable, and do not allow duplicate values.

1. Unordered: The items in the set do not have a defined order.
2. Unchanged: We cannot change items after the set has been created.
3. Duplicates not allowed: Set cannot have two items with the same value.

Methods in a set: -

add (), clear (), copy (), difference (), difference update (), discard (), intersection (), intersection update (), is disjoint (), is subset (), is superset (), pop (), remove (), symmetric difference (), union (), update ().

Dictionary: -

Dictionary items are ordered, changeable, and don't allow duplicates.

Methods in dictionary:

clear (), copy (), from keys (), get (), items (), keys (), pop (), pop item (), set default (), update (), values ().

EXPERIMENT-2

Date:25-07-23

1. Write a program to control the VACCUM CLEANER moves(Intelligent systems design process).

Program:

```
ER=input("Enter room name(A/B)")
RAS=int(input("Enter the status of roomA(0/1)"))
RBS=int(input("Enter the status of roomB(0/1)"))
if(ER=="A"):
  if(RAS==1):
    print("clean the roomA")
    RAS=0
    print("roomA is cleaned-shift right to B")
  if(RBS==1):
    print("clean the roomB")
    RBS=0
  print("claening process is completed")
if(ER=="B"):
  if(RBS==1):
    print("clean the roomB")
    RBS=0
  print("roomB is cleaned-shift left to A")
  if(RAS==1):
    print("clean the roomA")
    RAS=0
  print("claening process is completed")
```

OUTPUT:

Enter room name(A/B)A

Enter the status of roomA(0/1)1

Enter the status of roomB(0/1)1

clean the roomA

roomA is cleaned-shift right to B

clean the roomB

claening process is completed

Experiment-3

1.write a program to solve Monkey & Banana Problem.

Program:

```
monkey_height = int(input("enter monkey height:"))
box_height = int(input("enter box height:"))
banana_height = int(input("enter the height:"))

# Monkey moves the box
if box_height < banana_height:
    box_height += 1
    print("Monkey moves the box.")

# Monkey climbs the box
if monkey_height < box_height:
    monkey_height += 1
    print("Monkey climbs the box.")

# Monkey grabs the banana
if monkey_height == banana_height:
    print("Monkey grabs the banana.")
    print("Monkey successfully reached the banana!")
else:
    print("Monkey can't reach the banana.")
```

output:

enter monkey height:2

enter box height:1

enter banana height:3

Monkey climbs the box.

Monkey grabs the banana.

Monkey successfully reached the banana!