20/8/24

Mid-term presentation : (Review Paper)

- problem (AI policy problem)
- why it is relevant / interesting
- justification
- Presentation should contain
  - Intro
  - motivation
  - logistics
  - planning (work split b/w team members & Timeline)

Python

Dictionary - Index is whatever user defined, keys/values are user defined
Disadv : Indexes are not in sequential order

```
q = { }
q["tomato"] = 5
q[1] = 3
q[4] = "potato"
q["tortoise"] = "rabbit"
```

("tortoise" ↓ key, "rabbit" ↓ value)

```
>>> q
{'tomato': 5, 1: 3, 4: 'potato', 'tortoise': 'rabbit'}
```

Set : only has unique elements

```
a = [1, 1, 2, 3, 4, 4, 4, 5, 5]
b = set(a)
>>> b
{1, 2, 3, 4, 5}
```

Tuple - Circular Brackets - You cannot change values in Tuple
- Alteration/modification not possible.

```
x = (1, 2, 3)
x[1] = 5  ⇒ Error
```

loops : used to do same task repeatedly. - Iteration

for loop

```
a = [1, 2, 3, 4, 5, 6]
for x in a :
    print(x)
```

o/p

1
2
3
4
5
6

```
for x in range (5, 11):
    print(x)
```

o/p :

5
6
7
8
9
10

Range : Multiplication Table example

```
x = range (5, 11)
for x in range (0, 10):
    for y in range (0, 10):
        print (str(x) + " x " + str(y) + " = " + str (x * y))
        print ("\n") .   => for newline.
```

\n => for newline

\t => for tab => bigger space

continue => go back to next iteration

```
x = 10
for i in range(0, 100) :
    if i % x == 0 :
        continue
    else :
        print (i)
```

o/p : * print 0 to 99 except 10, 20, 30 ... 90

break

```
x = 10
for i in range (1, 100):
    if i % x == 0 :
        break
    else
        print(i)
```

prints from 1 to 9

while loop → Runs till condition is True

```
x = 50
while True :
    print (x)
    x -= 1
    if x < 20 :
        break
```

o/p : prints from 50 to 20

```
x = 50
while x >= 20 :
    print (x)
    x -= 1
```

functions

```
>>> def avg ( x, y, z) :
        return (x+y+z)/3

>>> avg ( 10, 20, 30)
```

o/p : 20.0

Advantage :- Anything can be sent into a function - It may work or may not
Data Types doesn't matter

```
>>> avg ("cat", "dog", "bird")   => gives error

>>> def func1 (x, y, z) :
        a = x + y + z
        return a

func1 (10, 20, 30)   => gives 60
func1 ("cat", "dog", "bird") => gives 'catdogbird'
```

Recursion : function calling itself

fibonacci series : 1, 1, 2, 3, 5, 8, 13, 21, 34, 55 ...

using iteration (loops) : computation (processor) is consumed

```
x = 1
y = 1
while True :
    print (x + y)
    t = x + y
    x = y
    y = t
```

o/p : 2, 3, 5 } Infinite series

using recursion (functions) : Memory is consumed - fill out RAM & crash

```
x = 1
y = 1
>>> def fib1 ( x, y ) :
        print (x + y)
        fib ( y, x + y )

>>> def fib ( x, y )
        print (x)
        print (y)

>>> fib(1, 1)
```

o/p : 1 1 2 3 ....

* Pip - Premade libraries

Numpy => library for mathematics

scipy => For algorithms, calculation. eigen values, algebra

matplotlib => for visualization -> graphs

scikit learn => end to end ML library => provides basic ML functionalities

nltk

keras, lasagna => Wrappers.

* Nltk book => Natural language processing toolkit

=> first 8 chapters.

* NLP - subarea of ML deals with processing, understanding of language

1) classification : we will give text, it will classify verb, pronoun, noun } Parts of speech Tagging

This is not cricket } can be noun

⇓

fair } can be adjective.

2) Segmentation : Break into sentence ~~into words~~

- using space

- Then what about wouldn't, shouldn't ?

3) Tokenization : Break sentence into words

4) Tagging - Part of speech Tagging. Name entity Tagging

5) ENtity Detection :

named entity

start label, stop label

6) Relation Detection :

- Relation between two phrases

- which two words relate to same thing

* Stanford coreNLP website.