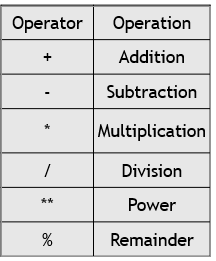
## Notes

### Python operators



### Other operators: // **floor division:** it divides and discard the reminder

Example: - 24//5 == 4

* But 24/5==4.8

% **modulo**: gives the reminder

Example: - 23%5 ==3

### Operations on Strings

The only operations that work on string are **+** and **\***

 + : adding  on string concatenates on the string

    Example: "a" + "b" == "ab"

\* : multiplying a string by a number repeats the String

example: "ab" \* 3 == "ababab"

Python types:

The main types we use in this session are , string, integer and Boolean: but python have other types for that refer the slides

To check the type of stg in python use type(), function

Type conversion :

* You can also use int() and float() to convert between strings and integers
* You will get an error if the string does not contain numeric characters

### python functions

return vs print

print just print the value on the console, while returns rgives us the value:

for example

def add\_print(a, b):

    print(a+b)

def add\_return(a, b):

    return a+b

the difference b/n the 2 functions is

add\_print() -> return value is void, while

add\_return() 🡪 return value is **int** or **string** depending on the input

so for example if there is another function that the above functions to determine the sum of a two numbers is greater than 10, the output will be different:

#  this allways returns False

def is\_greater\_than\_10(a, b):

    a = add\_print(a, b)

    if a> 10:

        return True

    else:

        return False

#this returns true if he sum is greater than 10

def is\_greater\_than\_10(a, b):

    a = add\_return(a, b)

    if a> 10:

        return True

    else:

        return False

# while the first one always return false, but if we use add\_return() it returns true if he sum is greater than 10

### Loops

2 types: **while** & **for** loops

**For loops**

# this prints 0,1,2,3,4

for x in range(5):

        print(x)

# this prints 2-6, starting from 2 -> 2, 3, 4, 5

for x in range(2, 6):

    print(x)

**Accumulator Loop: Summation**

# the total variable is to store the values, iterates and sums the numbers on to

# the total

total = 0

for n in range(1, 6):

   total = total + n

print(total)

**Lists**

* A List is a kind of Collection, A collection allows us to put many values in a single “variable”

Example

marks = [12, 15, 9, 12]

**we can iterate through lists**

for i in marks:

    print(i)

# this prints each mark

12

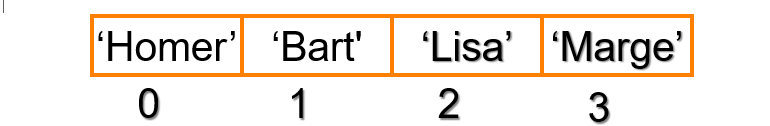
15

9

12

**Or we can get them with index**

simsons =[ 'Homer', 'Bart', 'Lisa' , 'Marge']



### Questions

1. Make a function that accepts an integer and determines, if it is leap year, the conditions for leap year are the following,
   1. A leap year must divide by 4,
   2. If it divides by 100, it must also divide by 400

Example:

40 is leap year

100: is not leap year, b/c it is divisible by 100, but not by 400

104: is leap year b/c it is only divisible by 4

400: is leap year

1000: is not leap year

**Code**

One way to do it

def leapYear(num):

    if num%4 ==0:  # first checking if it is divisible by 4

        if num % 100== 0:

            if num % 400 ==0:

                return True

            else:

                return False

        else: #if it is divisible by 4 and not divisible by 100, then it is a leap year

            return True

    else: # if not divisible by 4 it is not leap year at all

        return False

Cleaner way to do it

def leapYear(num):

    if num%4 ==0:  # first checking if it is divisible by 4

        if num % 100== 0 and num %400 !=0:

            return False

        else:

            return True

    else: # if not divisible by 4 it is not leap year at all

        return False

1. Write a function that takes a string , an index and a character and changes the character of that string at that value

def mutate(str, index, char)

EXAMPLE:

If

    str = abebe

    Index = 2

    Char = z

The output must be:  abzbe

If

    str = banana

    index = 0

    char = k

The output must be:  kanana

Thought processes

Because str is not a list we can not mutate it, ie strings are immutable, means we can not just say

Str[index]=char # this would give error

So we need to create a new string

#### Code implementation

def mutate(Str, index, char):

    new\_string= ""  # this is accumulator to acumulate the new string

    for i in range(len(str)):

        if index== i:   # checks whether index == the position we want to change

            new\_string+=char

        else:

            new\_string+=Str[i]

    return new\_string

these are the questions from your assigmnets

1. Counting digits

Code

def counting(n):

    cnt =0

    while n>0:

        n=n//10

        cnt +=1

    return count

frequency is basicly the same with counting with little modificarion

def frequency(num, c):

    cnt=0

    while num>0:

        val=num%10

        num=num//10

        if val==c:

            cnt+=1

    return cnt

big v

to draw the big v first lets make a function that makes just half v

 \

  \

   \

    \

     \

def bigV2(n):

    for i in range(n-1):

        print(" "\*i,"\\")

to make this a full V we just have to adjust the print statement

# the size of the v might vary but once you get the logic it is just a matter of adjusting the middle space

def bigV3(n):

    middle= " "

    middle\_space= n\*2 -4

    for i in range(n-1):

        print(" "\*i,"\\", middle \* middle\_space, "/")

        middle\_space-=2