if statement

a

>
<
>=
<=
!=
==

b

```
if condition :
      # True block
else:
      # False block
```

condition A

and
or

condition B

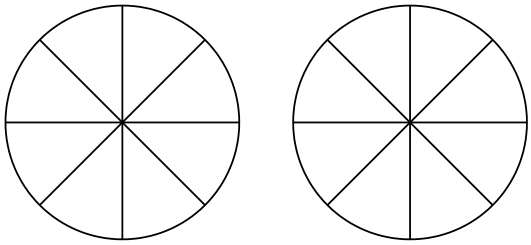
str

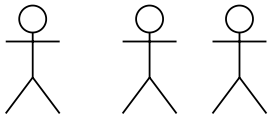
is,==,len(str),in

exercise 1 :
ask the user to enter his id number
check if the id number is valid

valid id number contain 9 digits

exercise 2:
fact: a pizza has 8 slices
you are planning a party, and you wan order pizza for your friends
you have a math issues so you decided to build python application to calc the number of slices each one gonna have and the left-overs,
write a python code that asks the user how many participants they invited
and how many pizza's ordered
the application must show the number of slices per person and the leftovers
and if the lift overs is exactly half of the slices per person number
print 'each one will get a 0.5 meal yaaaay'





2 pizzas
16 slices
5 slices per person
1 leftovers

exercise 3 :
ask the user to insert value from 0-2
0- for printing his name
1- for calculate his age
2- for printing the half of his address

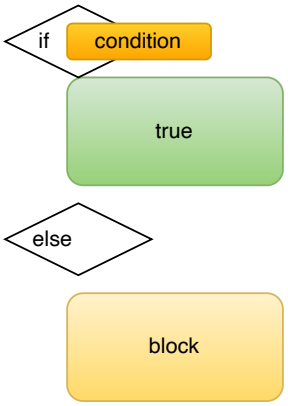
if the user chooses 0 then:
ask his name , save it in a var , print the name in title
if the user chooses 1 :
ask for his birth-year and calc his age please
if the user hit the 2 :
ask for his address and print half of it

example:

welcome to my terminal
i) if you wish to print your name here , press 0
ii) if you with to calc you age , press 1
iii) if you wish to print your address with bug , press 2
2
whats your address: Nof haaretz 1 , rosh haayen
Nof haaretz

elif
nested if
match

write a code that take the student grade and if the grade in
above 90 -> print A
if the grade between 90-80 print B
if the grade between 70-60 print C
otherwise F




```
if grade > 90 :
    print('A')
elif grade <= 90 and grade >80:
    print('B')
```

```
match var:
    case 1:
        # true block
```

list

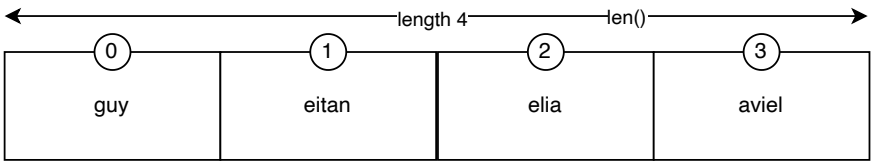
list

is a collection that can store
multipule values in one variable

-ordered
- mutable
- allow duplications

```
numbers = [1,2,3,4,5,6,88] # list of numbers
foods = ['kobeh','pizza','apple pie'] # list of strings
mix = [ 1 , 'eitan', True , [1,2,3] ]
```

names



Accessing Elements

```
print(Names[0]) # guy
print(Names[5]) # error index out of range
print(Names[-1]) # aviel negative index
```

change items

```
names = ['guy','eitan','elia','aviel']

names[2] = 'rotem'
print(names)
['guy','eitan','rotem','aviel']
```

add items

```
names = ['guy','eitan','elia','aviel']
name.append('yana')
print(names)
['guy','eitan','elia','aviel','yana']

names.insert(1,'sahar')
print(names)
['guy','sahar','eitan','elia','aviel','yana']
```

remove items

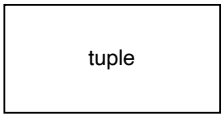
```
names= ['guy','eitan','elia','aviel']

names.pop(1) # delete by index
print(names)
['guy','elia','aviel']

names.remove("aviel") #delete by value
print(names)
['guy','elia']

del names[0]
print(names)
['elia']
```

```
names.sort()
names.reverse()
names.count()
names.index()
names.clear()
```

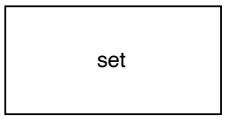


tuple

is a collection that can store
multipule values in one variable
but we cannot change them

-ordered
- immutable
- allow duplications

```
numbers = (1,2,3,4,5,6,88) # list of numbers
foods = ('kobeh','pizza','apple pie') # list of strings
mix = ( 1 , 'eitan', True , [1,2,3] )
```



SET

is a collection that can store
unique value only

-unordered
- mutable
- does not allow duplications

```
numbers = (1,2,3,4,5,6,88) # list of numbers
foods = ('kobeh','pizza','apple pie') # list of strings
mix = ( 1 , 'eitan', True , [1,2,3] )
```