Section 3

1. Write a Python program to remove the vowels from a list of strings and save the new string without vowels. Note: the vowels (A E I O U). [4M]

In [7]:
```
mylist=["Hello", "Hi", "Python", "Programming"]
vowels=['A', 'E', 'I', 'O', 'U']
wordDict={} #0.5 and adding value
for word in mylist: #1M
    num=0 #0.5 and increasing value
    for i in range(len(word)): #1M
        if word[i].upper() in vowels: #1M
            num+=1
    wordDict[word]=num

print(wordDict)
```

#anthor style of looping char by char into a list:

```
for word in mylist:
    num=0
    for char in word:
        if char.upper() in vowels:
            num+=1
```

```
{'Hello': 2, 'Hi': 1, 'Python': 1, 'Programming': 3}
```

1. Write a Python program to input a key and check whether a given key already exists in a dictionary and print “yes, the key exists” or “no, the key doesn’t exist” [2M]

In [11]:
```
d = {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
key=int(input("please enter the key:")) #0.5

if key in d: #1.5
    print("yes")
else:
    print("No")
```

```
please enter the key:8
No
```

1. Using While, write a function return_sublist(s) that takes in a list of numbers and returns the numbers in the list that appeared before a -1 in the list. For example, count_transitions([2,5,2,5,-1,6,7,7,8] should return [2,5,2,5] [4M]

In [6]:
```
def return_sublist(mylist): #1 and return
    i=0
    newlist=[] #1
    while mylist[i]!=-1: #1 and adding
        newlist.append(mylist[i]) #1
        i+=1
    return newlist

mylist=[2,5,2,5,-1,6,7,7,8]

print(return_sublist(mylist))

#bonus 1
i=-1
newlist=[]
while mylist[i]!=-1:
    newlist.append(mylist[i])
    i-=1
newlist.reverse()
print(newlist)
```

be carful , you are required to use the while loop

anthor style by using for loop for q1 :

```
def return_sublist(mylist):
    newlist=[]
    for num in mylist:
        if num==-1:
            break
        else:
            newlist.append(num)
    return newlist
```

```
[2, 5, 2, 5]
[6, 7, 7, 8]
```

1. Using the following methods that can be used with characters and strings (isupper(), islower(), lower(), upper()), Write a function flip_case(s) that takes in a string and returns a new string in which each character is flipped to its opposite case. [4]

In [1]:
```
def flip_case(s): #1 and return
    i=0
    newString="" #1 and appending
    for i in range(len(s)): #1
        if s[i].isupper(): #1
            newString+=s[i].lower()

        else: #1
            newString+=s[i].upper()
    return newString

mystring="ssTT"
print(flip_case(mystring))
```

anther style of soluation by looping without range(len (string))

```
def flip_case(s):
    newstring=""
    for i in s:
        if i.isupper():
            newstring+=i.lower()
        else:
            newstring+=i.upper()
    return newstring
mystring="ssTT"
print(flip_case(mystring))
```

```
SStt
```

Section 2: output

In [30]:
```
def car(x):
    print('in car, x is', x)
    y = boat(x - 1) + boat(x + 2)
    print('in car, y is', y)
    return y

def boat(y):
    print('in boat, y is', y)
    x = y * 2 + 3
    print('in boat, x is', x)
    return x


y = car(3)
print('at this level, y is', y)
```

```
in car, x is 3
in boat, y is 2
in boat, x is 7
in boat, y is 5
in boat, x is 13
in car, y is 20
at this level, y is 20
```

In [1]:
```
def quiz(x):
    print('x is', x)
    if x < 1:
        return 2
    else:
        p = 6 - (x - 1)
        print('p is', p)
        return p -1

y = quiz(3)
print('y is', y)

y = quiz(1)
print('y is', y)

y = quiz(0)
print('y is', y)
```

```
x is 3
p is 4
y is 3
x is 1
p is 6
y is 5
x is 0
y is 2
```

In [2]:
```
def see(x,y):
    return y,x

def M(x,y):
```

```
    if(x<y):
        x,y= see(x,y)
    return x-y

print(M(20,30))
print(M(8,3))
```

```
10
5
```

section1

1. print the value of 20 from tuple1.
2. evaluate the value of city[5:]
3. print the full name of “Aron” in d
4. copy elements 50 and 70 from tuple2 into a new tuple.
5. print the value “bee” in animal
6. print the reverse of city.
7. evaluate the value of myList[:2]

In [3]:

```
d = {"students":[{"firstName": "Nikki", "lastName": "Roysden"},
{"firstName": "Mervin", "lastName": "Friedland"},
{"firstName": "Aron ", "lastName": "Wilkins"}]}
city = 'Cape town'
myList = [3, 2, -1, [4, 7], 5]
tuple1 = ("Orange", [10, 20, 30], (5, 15, 25))
tuple2 = (10, 30, 50, 70, 90)
animal = {1:"lion", 2:"bee", 2:"cat"}
```

In [10]:

```
#1
print(tuple1[1][1])
#2
print(city[5:])
#3
print(d["students"][2]["firstName"],d["students"][2]["lastName"])
#4
t=tuple2[2:4]
print(t)
#5
#cannot be printed
#6
print(city[::-1])

#7
print(myList[:2])
```

print (d["students"][2]) #also true

```
20
town
Aron  Wilkins
(50, 70)
nwot epaC
[3, 2]
```

In [ ]: