Programming in Python Lecture 3- If statements, For loop, While loop

Plan for today

- If statements
- For loop
- While loop

Reminder

- Lists
- Tuples
- Dictionaries

Lists are Indexable

Remember this?

```
>>> str="We are learning Python!!!“
>>> str[1:3]
‘e ‘
>>> str[0:3]
‘We ‘
>>> str[1:]
"e are learning Python!!!’'
>>> str[-4:-2]
'n!’
>>> str[:-3]
'We are learning Python"
>>> str[-3:]
'!!!’
```

The same indexing + slicing works for lists!

Lists are Indexable

```
>>> list = [9,8,6,1,5]
>>> list[0]
9
>>> list[4]
5
>>> list[-3]
6
>>> list[::2]
[9,6,5]
>>> my_list[5]
Traceback (most recent call last):
File "<pyshell#7>", line 1, in <module>
my list[5]
IndexError: list index out of range
```

9	8	6	1	5
0	1	2	3	4
-5	-4	-3	-2	-1

Slicing

```
# reverse
>>>list[::-1]
[5, 1, 6, 8, 9]


# slicing does NOT change original list
>>>list
[9,8,6,1,5]
```

Lists – Dynamic

Maintain a list of the students either by name or by id:

```
>>> students = [‘Sharon’,9654520, ‘Liat’,837561405
  ‘Yaniv’,968554353]

>>> students[2]

‘Liat’
```

- Roze decided to join the course, so we update the list:

```
# append - add an element to the end of the list

>>> students.append(‘Roze')

>>> students

students = [‘Sharon’,9654520, ‘Liat’,837561405
  ‘Yaniv’,968554353,’Roze’]
```

Lists – Dynamic

- Yaniv wants to leave the course:

```
>>> students.remove('Yaniv')
>>> students
```

remove removes only **the first** occurrence of a value.

Assignments of List Variables

```
>>> list_1= [1,2,3]
>>> list_2= list_1
>>> list_1 = [6,7,8,9]
>>> list_2
[1,2,3]
>>> list_1
[6,7,8,9]
```

So far - no surprises

Assignments of List Variables

```
>>> list_2= list_1
>>> list_1[0] = 1000
>>> list_1
[1000,7,8,9]
>>> list_2
[1000,7,8,9]
```

Surprise!

Nested Lists

```
NL = [ [3, 7, 2],[6, 0, 1] ]
Print(NL[1])
>>>[6, 0, 1]
Print(NL[1][0])
>>> 6
len(NL)
>>> 2
```

Tuples are immutable lists

```
>>> list = [1,2,3]
>>> list [1]=10
>>> tuple = (1,2,3)
>>> tuple[1] = 10
Traceback (most recent call last):
  File "<pyshell#20>", line 1, in <module>
   my_tuple[1] = 10
TypeError: 'tuple' object does not support item assignment
```

Tuples are immutable lists

- Why use Tuples?
  - Faster than lists in certain operations
  - The interpreter enforces that they are not changed, and we’ll see why this is useful.

Tuples

A tuple is similar to a list, but it is immutable.

```
>>> B = (“Let", “It", "be") # definition
>>> B
(“Let", “It", "be")
>>> B[0]      # indexing
"Let"
>>> B[-1]     # backwords indexing
‘Be'
>>> B[1:2]    # slicing
(‘It')
```

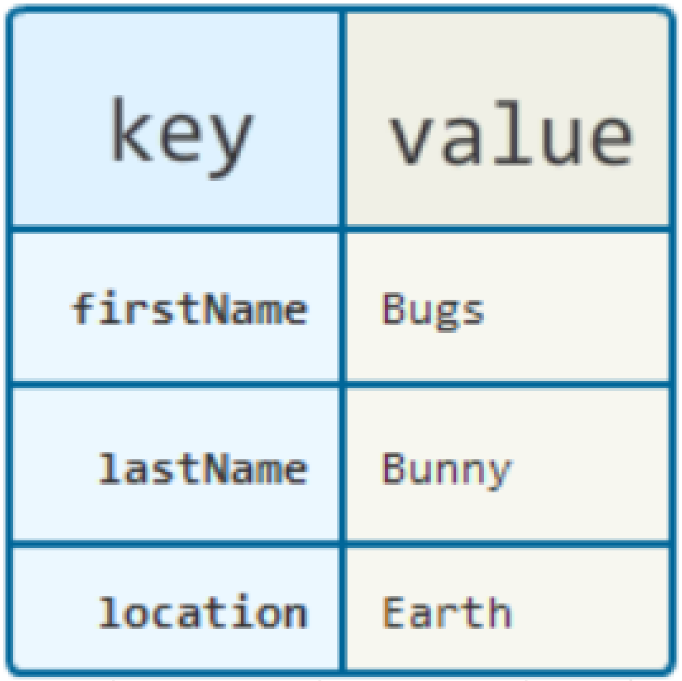
Tuples

```
>>> t[0] = 'do'  # try to change
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    t[0]='do'
TypeError: 'tuple' object does not support item assignment
```

No append / extend / remove in Tuples!

Dictionaries

- Key – Value mapping
  - No order
- Fast!
- Usage examples:
  - Database
  - Dictionary
  - Phone book



key	value
firstName	Bugs
lastName	Bunny
location	Earth

Dictionaries

Example: ID list- Map names to IDs:

```
>>> ID_list = {'Eric': ‘30145', ‘Shlomi': ‘38171',
  ‘Kobi': ‘85736'}
>>> print(ID_list)
{'Eric': ‘30145', ‘Shlomi': ‘38171', ‘Kobi':
  ‘85736'}
```

Note:The pairs order changed!

Dictionaries

Access dictionary Items:

```
>>> ID_list ['Eric′]
'30145'
```

Add a new person:

```
>>> ID_list[‘David'] = ‘84759'
>>> print(ID_list )
{'Eric': ‘30145', ‘Shlomi': ‘38171', ‘Kobi':
  ‘85736′, ‘David′:‘84759' }
```

Dictionaries

What happens when we add a key that already exists?

```
>>> ID_list [‘David']= ‘75647'

>>> print(ID_list)

{'Eric': ‘30145', ‘Shlomi': ‘38171', ‘Kobi':
  ‘85736’, ‘David’:‘75647 ' }
```

How can we add another Kenny McCormick in the phone book?

Homework

1. What is the fifteenth letter of the alphabet?
2. What is the code for the twenty-third letter of the alphabet?
3. What is the fourth letter of the code for the eighth letter of the alphabet?

Input:

```
>>> Alphabet = [["A", "Alfa"],  ["B", "Bravo"], ["C", "Charlie"],  ["D", "Delta"],["E", "Echo"],["F", "Foxtrot"],["G", "Golf"], ["H", "Hotel"], ["I", "India"],["J", "Juliett"],["K", "Kilo"],["L", "Lima"],["M", "Mike"],  ["N", "November"],["O", "Oscar"], ["P", "Papa"], ["Q", "Quebec“], ["R", "Romeo"],["S", "Sierra"],  ["T", "Tango"],  ["U", "Uniform"],["V", "Victor"], ["W", "Whiskey"],["X", "X-ray"],["Y", "Yankee"], ["Z", "Zulu"]]
```

Homework

4. Create a Python script that to Convert a given tuple of positive integers into an integer

Input:

```
>>>  nums=(1,2,3)
```

Output:

```
>>>  123
```

5. Create a Python script that change the first element in a tuple

Input:

```
>>> x = ("apple", "banana", "cherry")
```

Output:

```
>>>  ('kiwi', 'banana', 'cherry')
```

Homework

6. Create a Python script that combine values in python list of dictionaries.

Input:

```
>>>  item_list = [{'item': 'item1', 'amount': 400}, {'item': 'item2', 'amount': 300}]
```

Output:

```
>>>  {'item1': 400, 'item2': 300}
```

Plan for today

- **If statements**
- For loop
- While loop

Algorithms and Pseudo Codes-1

How do I get to work in the morning?

1. Get up
2. Drink coffee if my machine is working
3. Get out of the house
4. Walk for 10 minutes to the bus station
5. While waiting for the bus:
    - making a tiktok video
    - learn Python
6. Get on the bus

Think First, Code Later

How can I get to work in the morning?

1. Get up
2. Drink coffee **if** my machine is working
3. Get out of the house
4. Walk **for** 10 minutes to the bus station
5. **While** waiting for the bus:

    making a tiktok video

    learn Python

6. Get on the bus…….

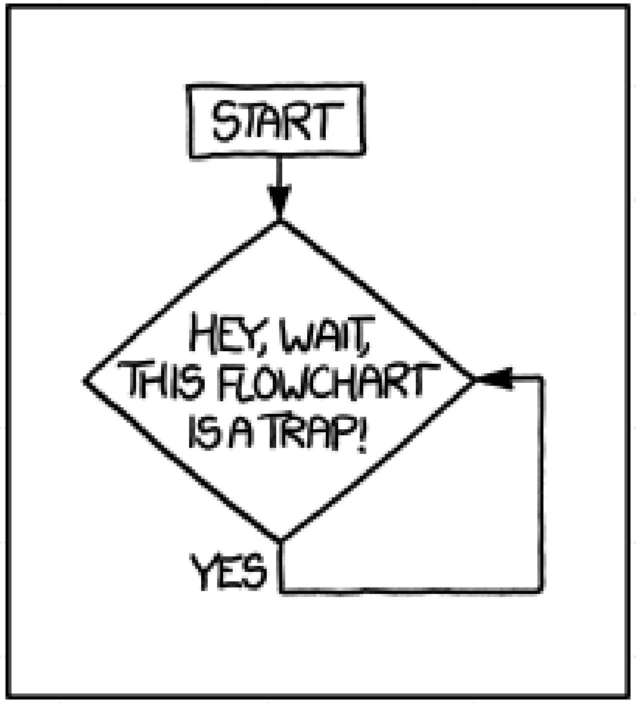
Flow Control

Different inputs →Different execution order

- Computer games
- Illegal input

Control structures

- **if-else**
- **for loop**
- **while** loop



http://xkcd.com/1195/

Conditional Statement: if

Used to execute statements conditionally

<u>Syntax</u>

```
if condition:
    statement1
    statement2
```

*If condition is **True**, statements are executed*

***Condition** = expression that evaluates to a Boolean*

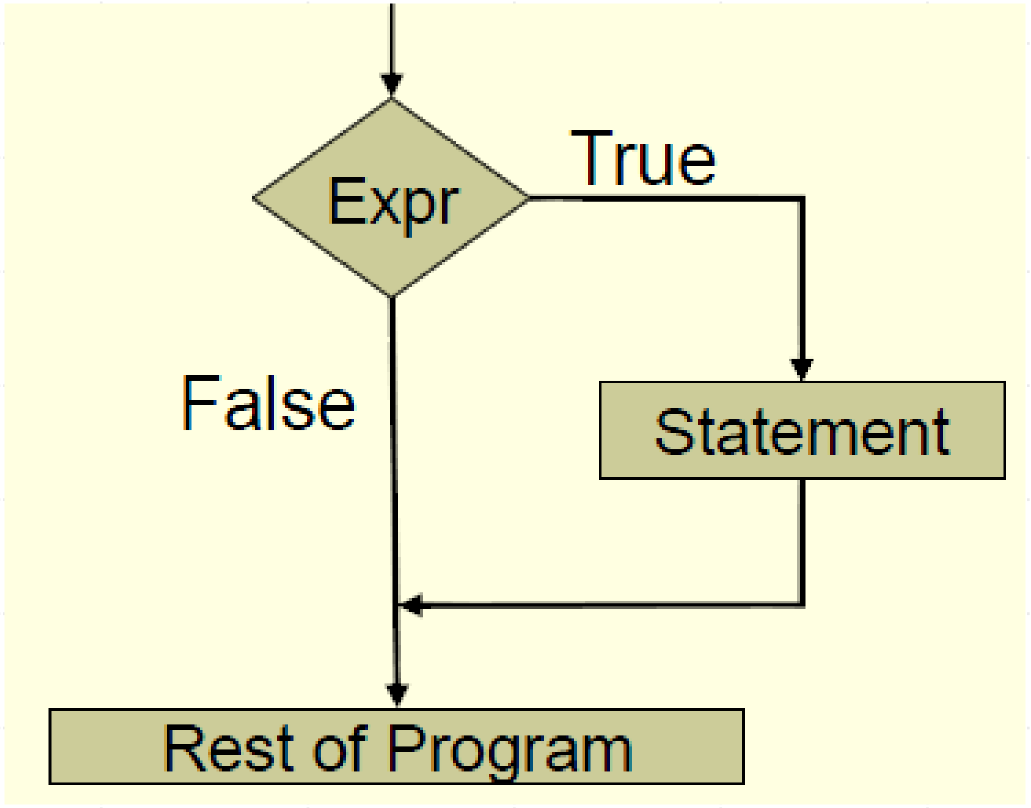
Indentation:

Following the if statement:

Open a new scope = one tab to the right.

Indicates the commands within the scope of this if.

Conditional Statements



elif

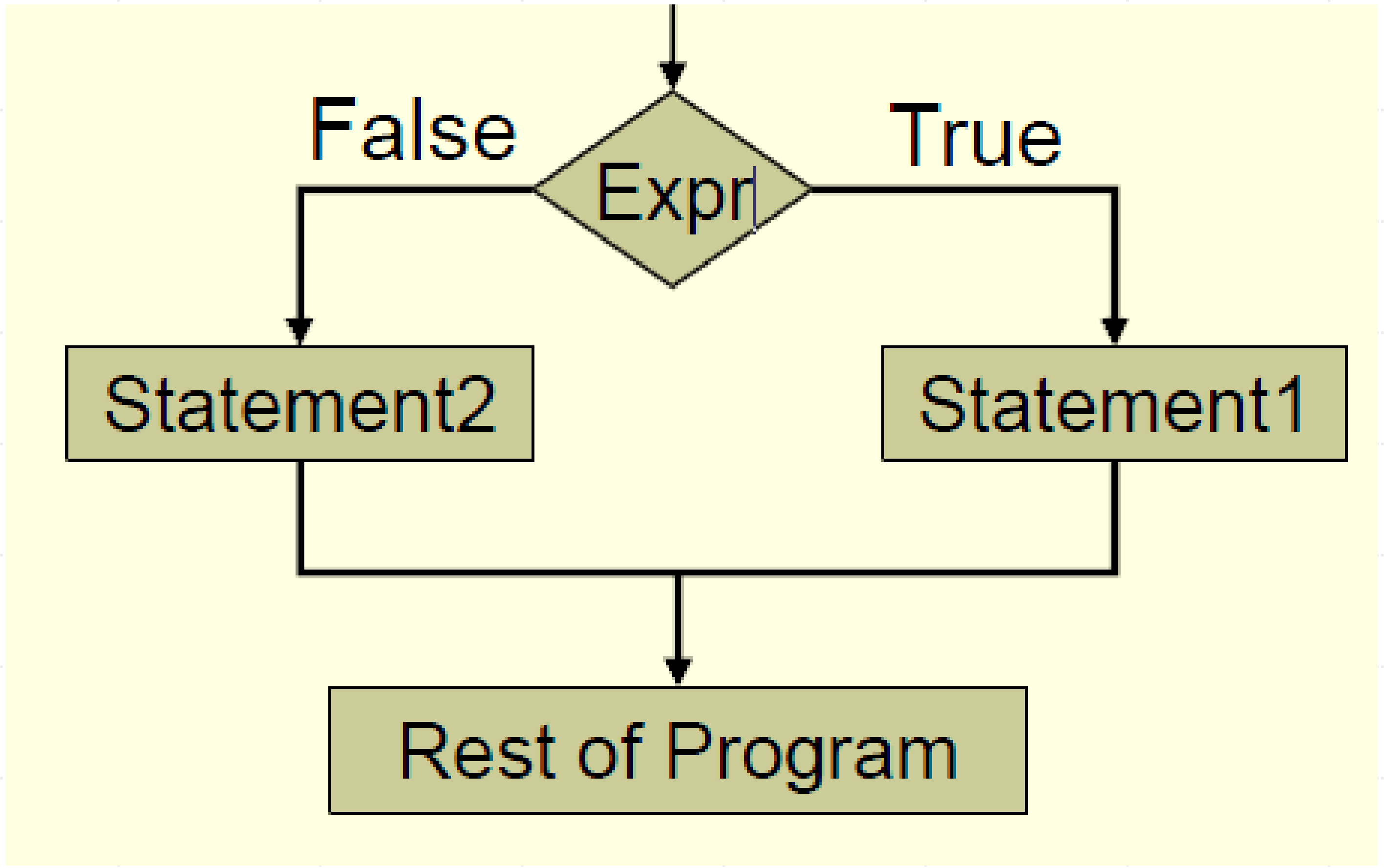
```
if  condition1:
   statement1
elif condition2:
   statement2
else:
   statement3
```

condition1 is true → execute ***statement1***

condition1 false and **condition2** true → execute ***statement2***

both conditions are false → execute ***statement3***

elif



Questions?

Pseudo Codes

- https://www.youtube.com/watch?v=Q3XWgQUq7zQ
  https://www.youtube.com/watch?v=9YRjX3A_8cM

Hands On

Plan for today

- **If statements**
- **For loop**
- While loop

For Loop

```
for element in iterable:
   statement1
   statement2
      …
```

Run over all elements in the object (list, string, etc.)

Iteration 0: Assign element = object[0]

- Execute the statements

Iteration 1: Assign *element* = object*[1]*

- Execute the statements

…

For Loop

Indentation determines the scope of the iteration.

Note

No infinite lists in Python !!!

No infinite for loops!!!

For Loop and Strings

Iterate over strings:

```
name = “Klay"
for letter in name:
    print "Give me", letter
print "What did we get?", name
```

```
Give me K
Give me l
Give me a
Give me y

What did we get?
```

Break – breaking loops

break terminates the nearest enclosing loop, skipping the code that follows the break inside the loop.

Used for getting out of loops when a condition occurs.

Example:

```
for elem in lst:
  if elem < 0:
      print "First negative number is", elem
      break
```

Continue

The continue statement, continues with the next iteration of the loop.
Example - create a list of unique elements:

```
>>> lst = [1,4,5,8,3,5,7,1,2]
>>> uniques = []
>>> for x in lst:
         if x in uniques:
               continue
         uniques.append(x)

>>> print uniques
[1,4,5,8,3,7,2]
```

Range

An ordered list of integers in the range.

```
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

range(from, to) contains all integers k satisfying from ≤ k **<** to.

range(to) is a shorthand for range(0, to).

```
>>> range(2,10)
[ 2, 3, 4, 5, 6, 7, 8, 9]
>>> range(-2,2)
[-2, -1, 0, 1]
>>> range(4,2)
[]
```

Range

```
>>> type(range(3))
<type 'list'>
```

Step size:

`range(from, to, step)` returns:

from, *from+step*, *from+2*step*,…, *from+i*step*

until *to* is reached, not including to itself.

```
>>> range(0,10,2)
[0, 2, 4, 6, 8]
>>> range(10,0,-2)
[10, 8, 6, 4, 2]
```

Range

```
>>> range(0, 10, -1)
[]
>>> range(0,10,0)
Traceback (most recent call last):
  File "<pyshell#21>", line 1, in <module>
    range(0,10,0)
ValueError: range() step argument must not be zero
```

Questions?

Hands On

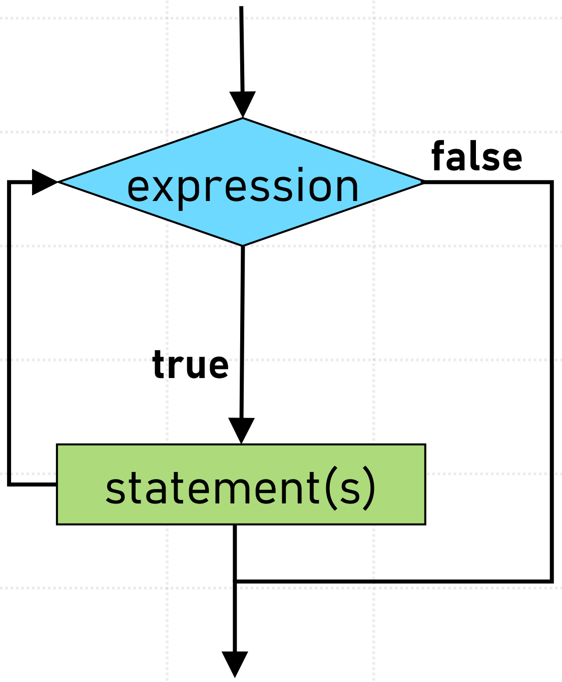
Plan for today

- **If statements**
- **For loop**
- **While loop**

While Loop

Used to repeat the same instructions until a stop criterion is met

```
while expression:
  statement1
  statement2
  …
```



Example - factorial

```
#factorial
n  = 9

fact = 1
i = 1
while i <= n:
     fact = fact * i
     i = i + 1
print n, "! = ", fact
```

- Can the `while` loop above be infinite?

Infinite Loops

```
i = 1
while i < 4:
 print i
```

Questions?

Hands On

for or while?

- In **most** cases it is more natural to use for
- In some cases it is better to use while
- for:
  - Predefined number of iterations
  - No need to initialize or advance the loop variable
- while:
  - Unknown number of iterations
  - Can specify a stop condition

Summary

- If statements
- For loop
- While loop

Homework

1. Create a Python program to get the largest/smallest number from a list

Input:

```
>>> s=[2,5,7,3,4,6]
```

Output:

```
>>>  The max number is: 7
>>>  The min number is: 2
```

2. Create a Python program to count the number of strings where the string length is 2 or more and the first and last character are same from a given list of strings

Input:

```
>>> words=['drd','1435','savg','11','sys','1321','10934','121’]
```

Output:

```
>>> ['drd', 'sys', '1321', '121']
```

Homework

3. Create a Python program to count all the names that start with "M" from a given list

Input:

```
>>>  names = ['Mor','Yuval', 'Many','Eli','Moshe']
```

Output:

```
>>>  3
```

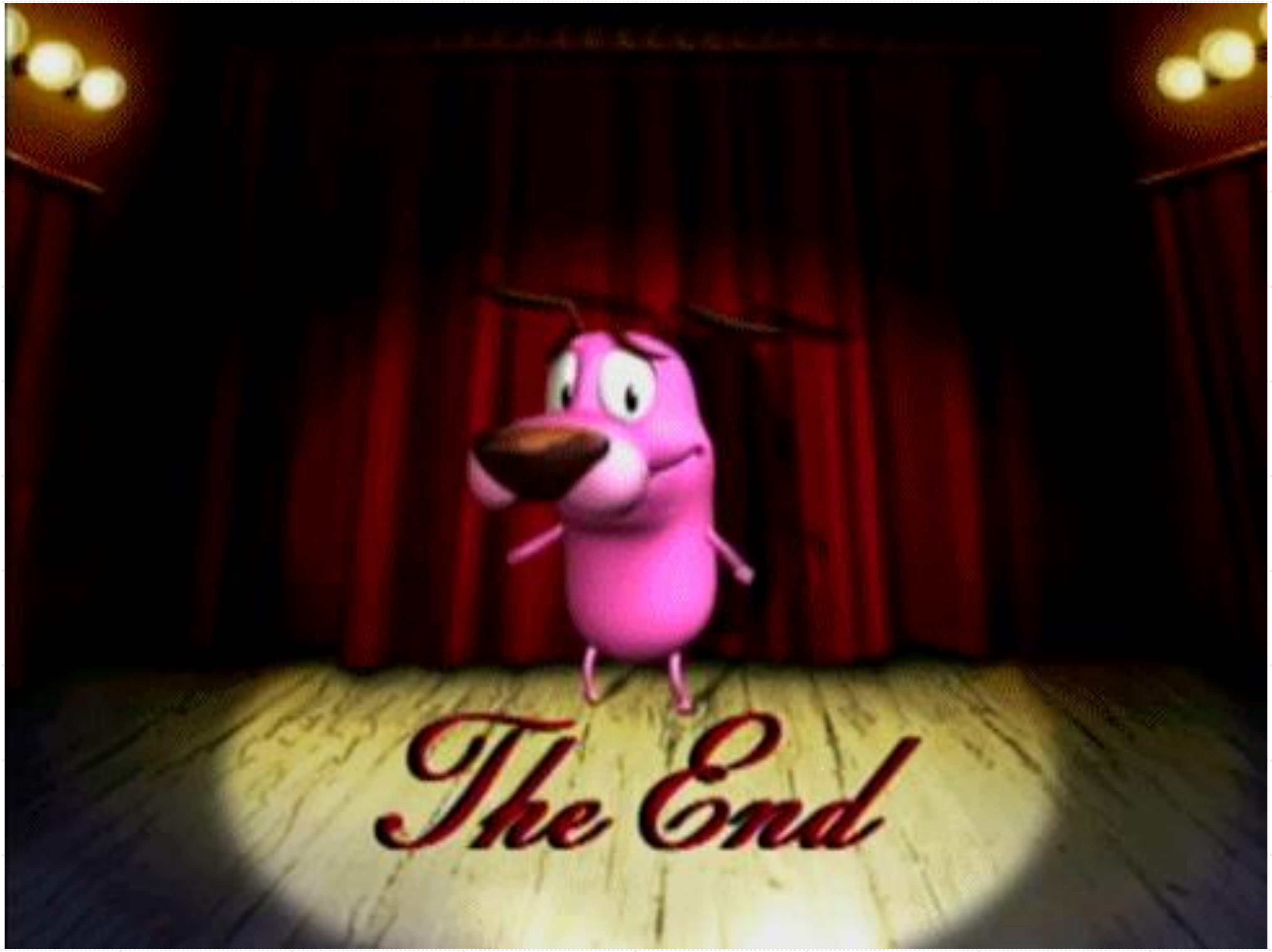
4. Create a Python script to calculate to price of Apple, Milk and Meat

Input:

```
>>>  Supermerket_list={"Apple":10,"Eggs":5,"Milk":5,"Bread":3,"Meat":20}
```

Output:

```
>>>  35
```


The End