1 Introduction

```
my_number = 10
print(type(my_number))
my_number = 10.0
print(type(my_number))
my_number = "10"
print(type(my_number))

my_number = "ten" + "hello"
print(my_number)
```

Listing 1: Python example

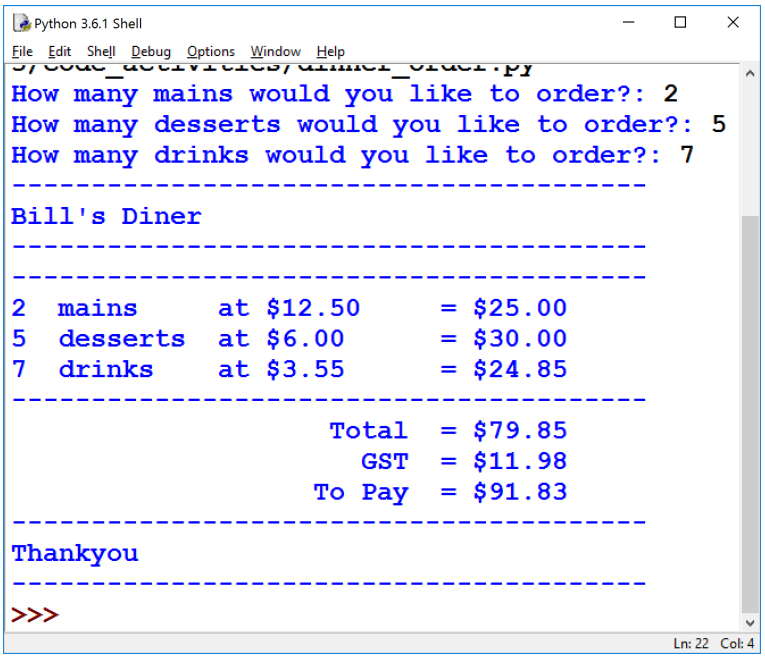
```
Python 3.7.2 (v3.7.2:9a3ffc0492, Dec 24 2018, 02:44:43)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======== RESTART: /Users/Paul/Desktop/code_activities/first_class.py ========
<class 'int'>
<class 'float'>
<class 'str'>
tenhello
>>>
```

Output

2 Basic Program

2.1 Dinner Order

You need to create a program that allows a waiter to enter in the number of main meals (at $12.50 each) , the number of desserts (at $6.00 each) and the number of drinks (at $3.55) ordered at a table.
The program then prints out a summary of the order, a total and then adds on GST (and gives the new total).
(Note: the program should be easy to alter so the price values should be declared as constant variables)



An example of Dinner Order program run

2.1.1 Planning

For the writing and planning:

- **Decomposition** Please write down your task decomposition.
  (This is breaking the task down into smaller pieces that will be combined to make the finished program)
  See examples below.

- **Version Log**
  Your version log should go here. Annotated screenshots are a good idea at this point

- **Component Trialling**
  Show that you have trialled each component here.
  You should also include notes that justify the major decisions you made.

- **Assembled Outcome Testing**

  Please show testing for your assembled outcome below.

  This should include a test plan followed by screenshot proof

- **Usability Testing**

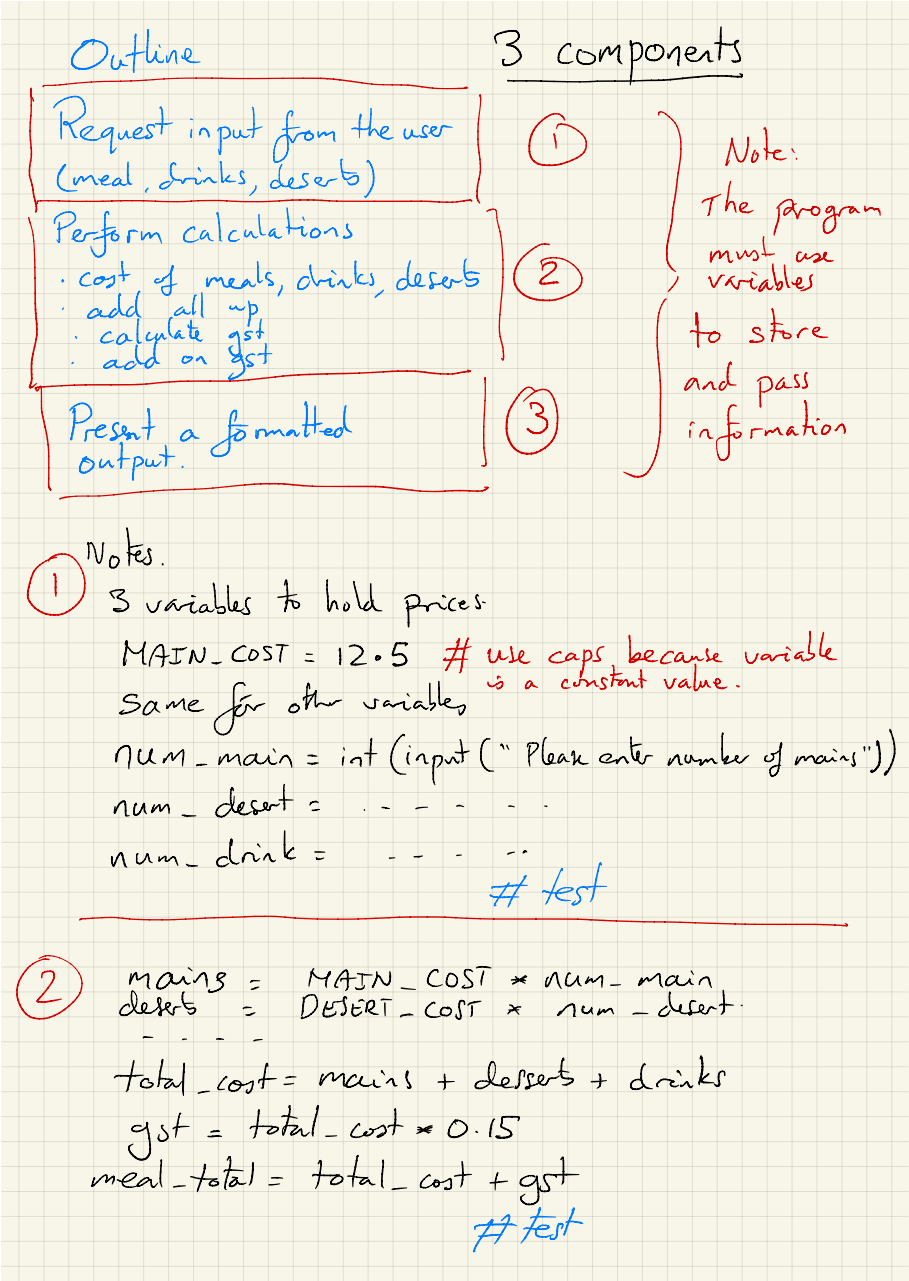
  Try the program out on someone else and make notes or video etc. Write a list of things improvements which need to be made based on your usability testing. Then write down what you changed.

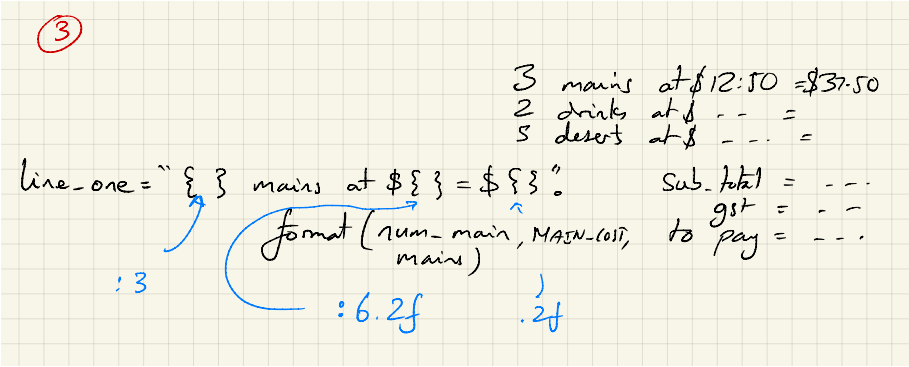
- **General Evaluation:**

  How good is may program? What can I do to improve it?

Examples of Component Planning

Basic Idea of Testing When testing we consider



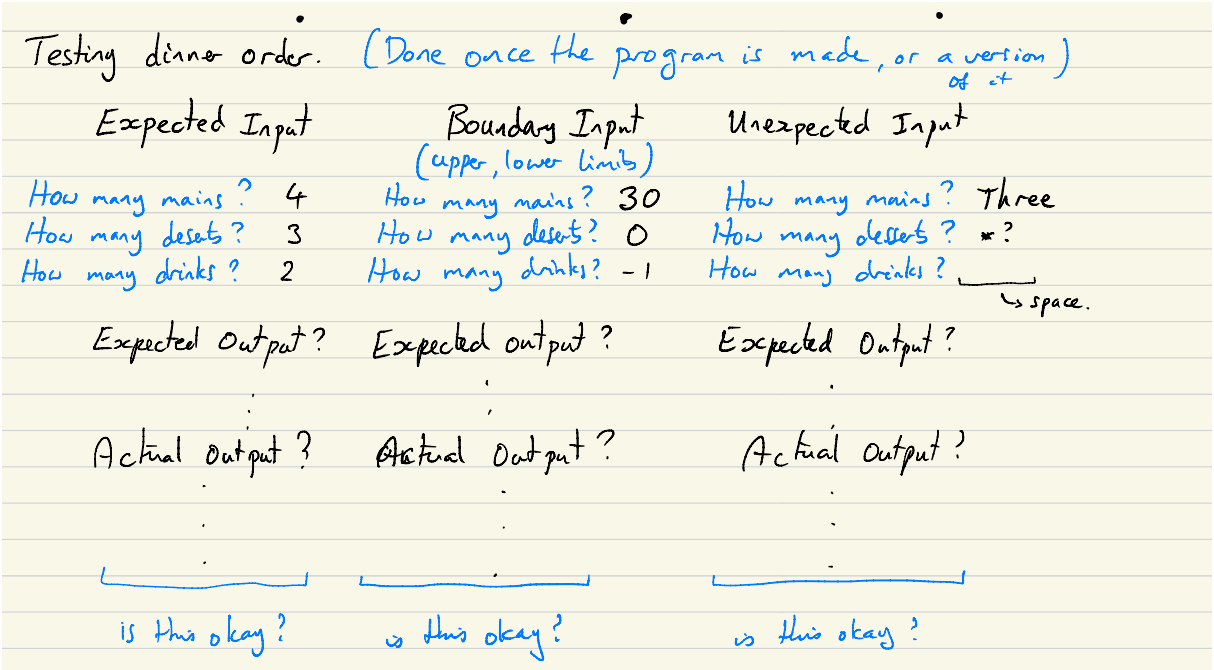


- Expected inputs (these are inputs we would normally expect the user to input).
  Does the porgram give the right outputs with the given expected inputs.

- Boundary inputs (are there maximum or minimum value that the user can enter)
  What happens at these boundaries (and if we go over them)

- Unexpected Inputs (these are character entries that we are not expecting)
  These could be just pressing enter or having spaces , using letters instead of numbers or characters like * , &, etc.
  These could occur if the user makes a mistake or misunderstands what is expected.

Testing should have some kind of plan and documentation of the results.



Result from expected

```
>>>
= RESTART: /Users/Paul/Desktop/code_activities/tr
How many mains would you like to order?: 4
How many desserts would you like to order?: 3
How many drinks would you like to order?: 2
----------------------------------------
Bill's Diner
----------------------------------------
4  mains     at $12.50     = $50.00
3  desserts  at $6.00      = $18.00
2  drinks    at $3.55      = $7.10
----------------------------------------
                   Total  = $75.1
                     GST  = $11.26
                  To Pay  = $86.36
----------------------------------------
Thankyou
----------------------------------------
>>>
```

Result from boundary

```
>>>
= RESTART: /Users/Paul/Desktop/code_activities/
How many mains would you like to order?: 30
How many desserts would you like to order?: 0
How many drinks would you like to order?: -1
----------------------------------------
Bill's Diner
----------------------------------------
30 mains     at $12.50     = $375.00
0  desserts  at $6.00      = $0.00
-1 drinks    at $3.55      = $-3.55
----------------------------------------
                   Total  = $371.45
                     GST  = $55.72
                  To Pay  = $427.17
----------------------------------------
Thankyou
----------------------------------------
>>>
```

Result from unexpected

```
>>>
= RESTART: /Users/Paul/Desktop/code_activities/trinket/dinner_order_basic.py =
How many mains would you like to order?: Three
Traceback (most recent call last):
  File "/Users/Paul/Desktop/code_activities/trinket/dinner_order_basic.py", line 1
3, in <module>
    mains = int(input("How many mains would you like to order?: "))
ValueError: invalid literal for int() with base 10: 'Three'
>>>
```

Evaluation

- What works
  - Works properly on expected inputs
  - Presentation of reciept clearly laid out on console.
- What could be improved
  - Should have an upper limit on how many meals can be entered
  - Should not allow entries below zero
  - Unexpected inputs cause a program crash
  - At the moment the program only runs once (so need a way to enter a new order)

Components: (parts of the program we need to be able to make).
(The components will then be assembled to make the finished program).

① Getting the user input
② Calculating the value of the meal
③ Printing out the receipt.

① (variable; polite, clear message to user.)

```
num_mains = int(input("How many mains would you like?"))
num_deserts = int(input("... desert ... ... ..."))
num_drinks = int(input("... ... drinks ... ... ..."))

print(num_mains)
print(num_deserts)
print(num_drinks)
```

← print out to test that it is working

Save with logical name: dinner_input_component

② Declare the prices as variables. UPPER CASE means will not change (constant).

```
MAIN = 12.5
DESERT = 6.0
DRINK = 3.55
```

just give the variables values for trialling purposes.

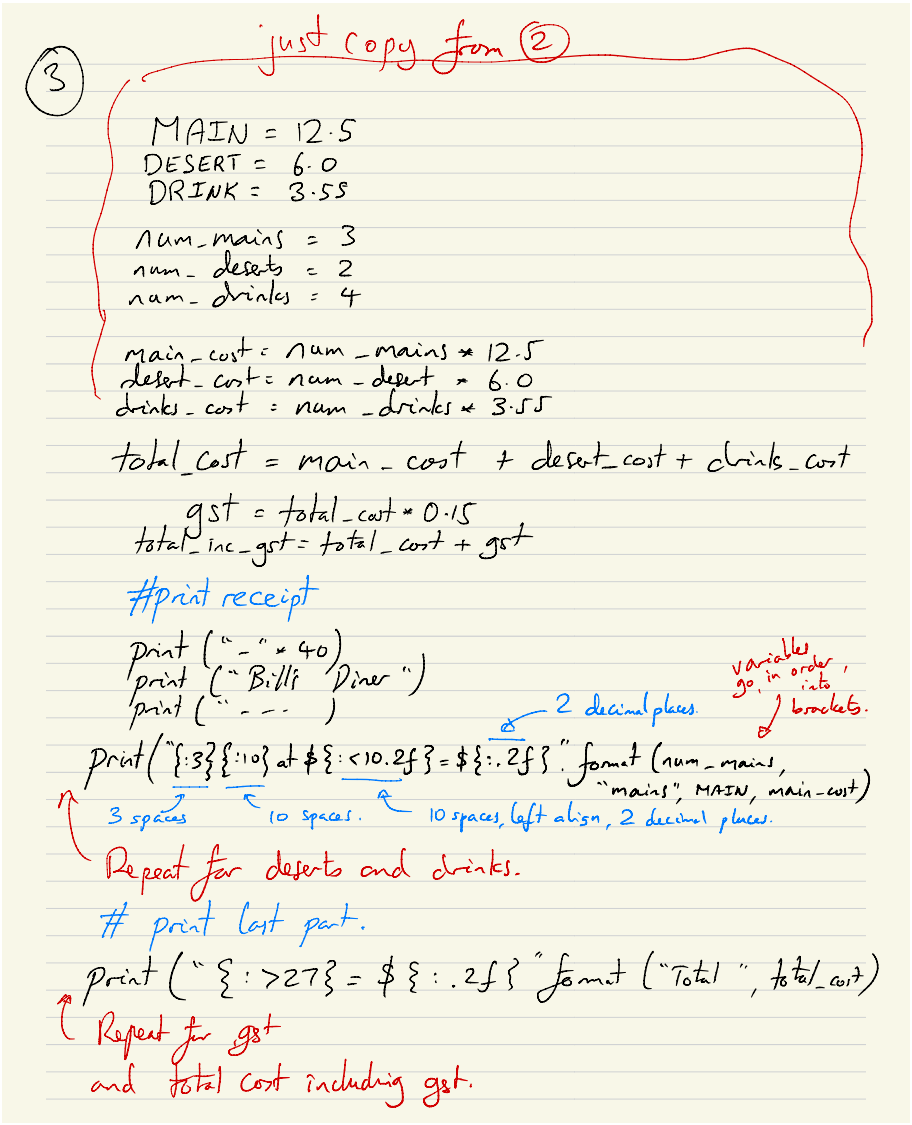
```
num_mains = 3
num_deserts = 2
num_drinks = 4

main_cost = num_mains * 12.5
desert_cost = num_deserts * 6.0
drinks_cost = num_drinks * 3.55

total_cost = main_cost + desert_cost + drinks_cost

print(total_cost)
```

← test.

```
gst = total_cost * 0.15
total_inc_gst = total_cost + gst
```

print as well

Save with a logical name: dinner_cost_component




```python
#constants
MAIN_PRICE = 12.5
DESSERT_PRICE = 6.0
DRINKS_PRICE = 3.55

#totalling variables (initially declared as 0)
mains_total = 0.0
desserts_total = 0.0
drinks_total = 0.0
ext_gst = 0.0

# input variables
mains = int(input("How many mains would you like to order?: "))
desserts = int(input("How many desserts would you like to order?: "))
drinks = int(input("How many drinks would you like to order?: "))

#calulate costs
mains_total = mains * MAIN_PRICE
```

```
desserts_total = desserts * DESSERT_PRICE
drinks_total = drinks * DRINKS_PRICE

ext_gst = mains_total + desserts_total + drinks_total
my_gst = ext_gst*0.15
my_to_pay = ext_gst + my_gst

print("-"*40)
print("Bill's Diner")
print("-"*40)
print("{:<3}{:10}at ${:<10.2f}= ${:.2f}".format(mains,"mains", MAIN_PRICE,
    mains_total))
print("{:<3}{:10}at ${:<10.2f}= ${:.2f}".format(desserts,"desserts",
    DESSERT_PRICE, desserts_total))
print("{:<3}{:10}at ${:<10.2f}= ${:.2f}".format(drinks,"drinks", DRINKS_PRICE,
    drinks_total))
print("-"*40)
print("{:>27}= ${}".format("Total  ", ext_gst))
print("{:>27}= ${:.2f}".format("GST  ", my_gst))
print("{:>27}= ${:.2f}".format("To Pay  ", my_to_pay))
print("-"*40)
print("Thankyou")
print("-"*40)
```

Listing 2: Dinner Order Program

```python
# get number of punnets boxes buckets
my_punnets = float(input("How many punnets?: "))
my_boxes = float(input("How many boxes?: "))
my_buckets = float(input("How many buckets?: "))

#testing
#my_punnets = 2.5
#my_boxes = 1.75
#my_buckets = 3.5

punnet_weight = my_punnets*0.25
box_weight = my_boxes*1.5
bucket_weight = my_buckets*7

total_weight = punnet_weight + box_weight + bucket_weight

total_price = total_weight * 5
print("."*30)
print("{:20}{:>6.2f} kg".format("Punnet weight" , punnet_weight))
print("{:20}{:>6.2f} kg".format("Box weight" , box_weight))
print("{:20}{:>6.2f} kg".format("Bucket weight", bucket_weight))
print("."*30)
print("{:20}{:>6.2f} kg".format("Total weight", total_weight))
print("."*30)
print("."*30)
print("{:20} ${:>6.2f}".format("Price to charge", total_price))
print("."*30)
```

Listing 3: Python example

3 Conditions

The solutions to many programming problems require an action to occur only if a particular condition is true.
A condition can only result in true or false.
(These values are called Booleans. True and False are reserved words in the Python language. True and False are values / objects of type bool)
A condition often involves an expression which compares one value with another.
Comparisons between numerical values are made using the same comparison operators that are used in mathematics:

$>$	greater than
$>=$	greater than or equal to
$<$	less than
$<=$	less than or equal to

and the equality / inequality operators:

$==$	equal to (the proper equals operator)
$!=$	not equal to

Boundary cases
Human languages are imprecise.
Does "every child over 5" mean "aged 5 and over" or "aged 6 and over"? The difference between $>=$ and $>$ is very important.
If you mean "every child aged 5 and over" you can express this as age > 4 or age $>= 5$.
Any lack of clarity needs to be removed at the design phase. Always clarify before coding. The boundary cases are where errors in programming often occur.
Boundary cases are the values at and just outside the specified limits.
In the case "aged 5 and over" the specified limit is 5 and the age values of 4 and 5 are the boundary cases.

conditions.py - C:/Users/khouripa/Documents/Year 11 programming/conditions.py (3.6.1)

File Edit Format Run Options Window Help

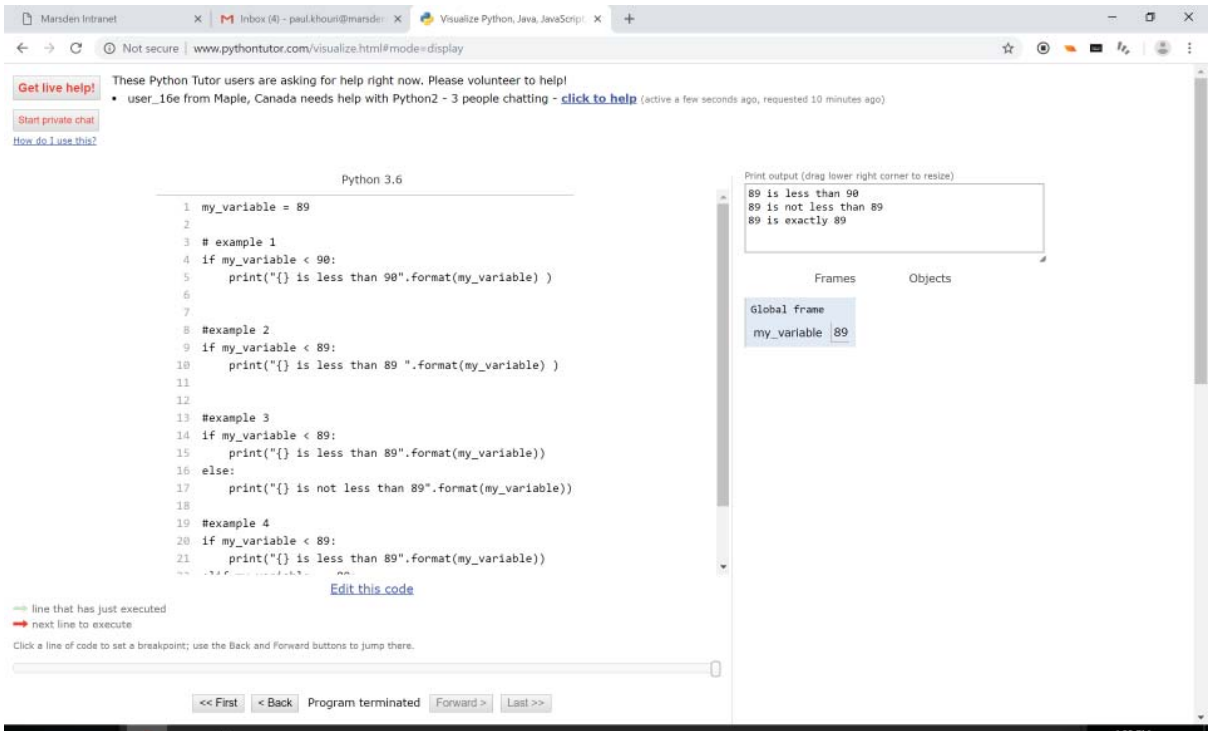
```
my_variable = 89

# example 1
if my_variable < 90:
    print("{} is less than 90".format(my_variable) )


#example 2
if my_variable < 89:
    print("{} is less than 89 ".format(my_variable) )


#example 3
if my_variable < 89:
    print("{} is less than 89".format(my_variable))
else:
    print("{} is not less than 89".format(my_variable))

#example 4
if my_variable < 89:
    print("{} is less than 89".format(my_variable))
elif my_variable == 89:
    print("{} is exactly 89".format(my_variable))
else:
    print("{} is more than 89".format(my_variable))
```

Only one condition can execute

Only one condition can execute

The code after the colon is "tabbed in" by 4 spaces.
The tabbed in code is what will run if the condition is met.

Running the code on pytutor : http://www.pythontutor.com/visualize.html



3.1 Activities

1. Consider a program that asks for a user's age and then:

   - if their age 12 or less, tells them that they should be at primary school,
   - if there age is more than 12 but less than 18, they need to go to secondary school
   - if they are 18 years old, it tells them that they have become an adult
   - and if they are older than 18 tells them that they must have left school.

2. To go to university a person needs to be:

   - over the age of 20
   - **or** be 16 or older and have level 3 NCEA
   - **and** (for both conditions) the person need to be competent with English.

   Create a program that given the following variables:
   age= 25
   ncea = False
   english = True
   Will give a correct response about whether a person can go to university or not.
   Test with other values.

4 Random numbers

```
# import the random number library
import random

#random number between 0 and 1
print(random.random())

# random number between two integers (inclusive)
print(random.randint(2,5))

# random number between two integers (inclusive)
# but with a step value (in this case will only get multiples
#of 5
print (random.randrange(0, 101, 5))

# create a list and randomly select a name from it
myList = ["Anna", "Michael", "Sarah", "Jessie"]
print(random.choice(myList))
```

Listing 4: Python example

5 Program using conditions and random numbers

```
import random
lower = 4
higher = 15
my_random = random.randint(lower,higher)
middle_number = round((lower + higher)/2)
#print(middle_number)
#print(my_random)

print("A random number has been chosen between {} and {}".format(lower, higher))
print(40*"-")
guess = input("Enter L if you think the number is less than {0} \n"
              "Enter G if you think the number is more than {0} \n"
              "Enter E if you think the number is equal to {0} : ".format(
   middle_number))

if guess == "L" and my_random < middle_number:
  print("you win")
elif guess == "G" and my_random > middle_number:
  print("you win")
elif guess == "E" and my_random == middle_number:
  print("you win")
elif guess != "G" and guess != "L" and guess != "E":
  print("you have not entered an appropriate letter")
```

```
23 else:
24   print("you lose")
25 print("The number was {}".format(my_random))
```

Listing 5: Python example

6 Loops

7 Lists

8 Functions

```
import random
def addition_question():
    a = random.randint(5,9)
    b = random.randint(7,10)
    my_sum = a + b
    print("{} + {} =".format(a,b), end="")
    return my_sum


question_number = addition_question()
answer = int(input(" "))
if answer == question_number:
    print("Correct")
```

Listing 6: Python example

```


my_string = "hello"

def test_function():
    global my_string
    my_string = "bob"

test_function()
print(my_string)
```

Listing 7: Python example

9 Dictionaries

```
car = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}

x = car.setdefault("model", "Bronco")

print(x)
```

```
10
11 quiz_one = {
12     "question" : "What is the capital of France? ",
13     "answer"   : "Paris"
14     }
15 quiz_two = {
16     "question" : "What is the capital of Spain? ",
17     "answer"   : "Madrid"
18     }
19 quiz_three = {
20     "question" : "What is the capital of Germany? ",
21     "answer"   : "Berlin"
22     }
23 print(quiz_one.get("question"))
24
25 user_answer = input( quiz_one.get("question") )
26 if user_answer == quiz_one.get("answer"):
27     print("Correct")
28 user_answer = input( quiz_two.get("question") )
29 if user_answer == quiz_two.get("answer"):
30     print("Correct")
31 user_answer = input( quiz_three.get("question") )
32 if user_answer == quiz_three.get("answer"):
33     print("Correct")
34
35 quiz_set = [
36     {
37     "question" : "What is the capital of France? ",
38     "answer"   : "Paris"
39     },
40     {
41     "question" : "What is the capital of Spain? ",
42     "answer"   : "Madrid"
43     },
44     {
45     "question" : "What is the capital of Germany? ",
46     "answer"   : "Berlin"
47     }
48     ]
49
50 for x in quiz_set:
51     user_answer = input( x.get("question") )
52     if user_answer == x.get("answer"):
53         print("Correct")
```

Listing 8: Python example

```
1 question_one = {
```

```
    "question": "What is the capital of France?",
    "A": "Paris",
    "B": "Moscow",
    "C": "Shanghai",
    "answer": "C"
    }
print(question_one["A"])
print(question_one["B"])
print("-"*30)

for x,y in question_one.items():
    if x in ["A", "B", "C"]:
        print("{} , {}".format(x,y))
```

Listing 9: Python example

10 Validation

```

high = 10
low = 1
error_int = "You have not entered an integer, please try again"
error_amount = ("You have not entered an appropriate amount\n"
                "it must be between $1 and $10")
msg = "Please choose between $1 and $10: "
get_value = False
while get_value == False:
    try:
        player_amount = int(input(msg))
    except ValueError:
        print(error_int)
        continue
    if player_amount < low or player_amount > high:
        print(error_amount)
        continue
    get_value = True
```

Listing 10: Python example

11 Coffee

```python
import random

char_list = ["C", "O", "F", "E"]



def run_trial():
    counter  = 1
    C_count = 0
    O_count = 0
    F_count = 0
    E_count = 0
    char_string = ""
    while True:
        my_char = random.choice(char_list)
        char_string += my_char
        # count number of C, O,
        C_count = char_string.count("C")
        O_count = char_string.count("O")
        F_count = char_string.count("F")
        E_count = char_string.count("E")
        if C_count >= 1 and O_count >= 1 and F_count >= 2 and E_count >= 2:
            break
        # increment count
        counter += 1
    #print(char_string)
    return len(char_string)


total_trials = 100
trials = 1
total_chars = 0
while trials <= total_trials:
    total_chars += run_trial()
    trials += 1


print(total_chars)
print(total_trials)

my_average = total_chars/total_trials
print(my_average)
```

Listing 11: Python example

```
import random

my_list=["c", "o", "f", "e"]

my_string = ""
count = 1
while count<100:
    my_letter = random.choice(my_list)
    my_string += my_letter
    #print(my_letter)
    if my_string.count("c")>=1 and my_string.count("o")>=1 \
       and my_string.count("f")>=2 and my_string.count("e")>=2:
        break
    count += 1
for x in my_list:
    print("{} * {}".format(x,my_string.count(x)))
print(my_string)
print("count is {}".format(count))
```

Listing 12: Python example

```
import random

my_list=["c", "o", "f", "e"]

my_string = ""
count = 1
while count<100:
    my_letter = random.choice(my_list)
    my_string += my_letter
    #print(my_letter)
    if my_string.count("c")>=1 and my_string.count("o")>=1 \
       and my_string.count("f")>=2 and my_string.count("e")>=2:
        break
    count += 1
for x in my_list:
    print("{} * {}".format(x,my_string.count(x)))
print(my_string)
print("count is {}".format(count))
```

Listing 13: Python example

```
import random

my_list=["c", "o", "f", "e"]

string_list = []

```

```
counter = 0
while counter < 1000:
    my_string = ""
    count = 1
    while True:
        my_letter = random.choice(my_list)
        my_string += my_letter
        #print(my_letter)
        if my_string.count("c")>=1 and my_string.count("o")>=1 \
           and my_string.count("f")>=2 and my_string.count("e")>=2:
            break
        count += 1
    string_list.append(my_string)
    counter += 1
    # print(my_string)
#print(string_list)
string_list.sort(key=len)
print(string_list[0])
print(string_list[- 1])

total_times = 0
number_trials = 0
for x in string_list:
    total_times += len(x)
    number_trials += 1
print(total_times)
print(number_trials)
```

Listing 14: Python example

```
import random

char_list = ["C", "O", "F", "E"]


chars_in_six = 0

C_count = 0
O_count = 0
F_count = 0
E_count = 0


trials = 0
trial_total = 100000
while trials < trial_total:
    counter  = 0
    char_string = ""
```

```
    while counter < 6:
        my_char = random.choice(char_list)
        char_string += my_char
        # count number of C, O,
        C_count = char_string.count("C")
        O_count = char_string.count("O")
        F_count = char_string.count("F")
        E_count = char_string.count("E")
        if C_count >= 1 and O_count >= 1 and F_count >= 2 and E_count >= 2:
            chars_in_six +=1
        # increment count
        counter += 1
    #print(char_string)
    trials += 1

print(chars_in_six/trial_total)
#print(char_string.count("C"))
```

Listing 15: Python example

```
import random

char_list = ["C", "O", "F", "E"]

# empty list
string_list = []


#----------------------------
def make_coffee():
    C_count = 0
    O_count = 0
    F_count = 0
    E_count = 0

    char_string = ""
    while True:
        my_char = random.choice(char_list)
        char_string += my_char
        # count number of C, O,
        C_count = char_string.count("C")
        O_count = char_string.count("O")
        F_count = char_string.count("F")
        E_count = char_string.count("E")
        if C_count >= 1 and O_count >= 1 and F_count >= 2 and E_count >= 2:
            string_list.append(char_string)
            break
        # increment count
```

```
29
30
31      #print(char_string)
32      #print( len(char_string) )
33      return len(char_string)
34  #--------------------------------
35
36
37  total_trials = 1000000
38  trials = 1
39
40  total_string_lengths = 0
41
42  while trials <= total_trials:
43      my_number = make_coffee()
44      total_string_lengths += my_number
45      #print(my_number)
46      trials +=1
47  #print(total_string_lengths)
48  average = round(total_string_lengths / total_trials)
49  print("On average you will need {} visits".format(average))
50  #print(string_list)
51
52  string_list.sort(key=len)
53  #print(string_list)
54  #print first item (thing)
55  print(string_list[0])
56  #print last item
57  print(string_list[total_trials-1])
58  print(    len( string_list[total_trials-1] ) )
59
60
61
62
63
64
65
66
67
68
69  #print(string_list)
70
71
72
73  #string_list.sort(key = len)
```

Listing 16: Python example

```
import random

char_list = ["C", "O", "F", "E"]

string_list = []


def run_trial():
    counter  = 1
    C_count = 0
    O_count = 0
    F_count = 0
    E_count = 0
    char_string = ""
    while True:
        my_char = random.choice(char_list)
        char_string += my_char
        # count number of C, O,
        C_count = char_string.count("C")
        O_count = char_string.count("O")
        F_count = char_string.count("F")
        E_count = char_string.count("E")
        if C_count >= 1 and O_count >= 1 and F_count >= 2 and E_count >= 2:
            string_list.append(char_string)
            break
        # increment count
        counter += 1
    #print(char_string)
    return len(char_string)




total_trials = 100
trials = 1
total_chars = 0
while trials <= total_trials:
    total_chars += run_trial()
    trials += 1


print(total_chars)
print(total_trials)
my_average = total_chars/total_trials
print(my_average)

#print(string_list)
```

```
string_list.sort(key = len)
#print(string_list)
print(string_list[0])
print(string_list[-1])
print(len(string_list[0]))
print(len(string_list[-1]))

my_dict = {}
for i in range(len(string_list[0]), len(string_list[-1])+1 ):
    my_dict[i]=0
    #print(i)

#print(my_dict)
for x in string_list:
    my_dict[len(x)]+=1
    #print(x)
print(my_dict)
my_sum = 0
for x in my_dict:
    my_sum += my_dict[x]
print(my_sum)
# dictionary not the same lenght as string list
for x,y in my_dict.items():
    temp = int(y)
    temp = temp/total_trials
    print(temp)
    my_dict[x]= temp
print(my_dict)
```

Listing 17: Python example

```
import random

char_list = ["C", "O", "F", "E"]


chars_in_seven = 0

C_count = 0
O_count = 0
F_count = 0
E_count = 0


trials = 0
trial_total = 1000000
while trials < trial_total:
    counter  = 1
```

```
    char_string = ""
    while counter <= 7:
        my_char = random.choice(char_list)
        char_string += my_char
        # count number of C, O, F, E
        C_count = char_string.count("C")
        O_count = char_string.count("O")
        F_count = char_string.count("F")
        E_count = char_string.count("E")

        # increment count
        counter += 1
    # test string at end
    if C_count >= 1 and O_count >= 1 and F_count >= 2 and E_count >= 2:
        chars_in_seven +=1
    #print(char_string)
    trials += 1

print(chars_in_seven/trial_total)
#print(char_string.count("C"))
```

Listing 18: Python example

```
import random

char_list = ["C", "O", "F", "E"]


chars_in_eight = 0

C_count = 0
O_count = 0
F_count = 0
E_count = 0


trials = 0
trial_total = 1000000
while trials < trial_total:
    counter  = 1
    char_string = ""
    while counter <= 8:
        my_char = random.choice(char_list)
        char_string += my_char
        # count number of C, O, F, E
        C_count = char_string.count("C")
        O_count = char_string.count("O")
        F_count = char_string.count("F")
```

```
        E_count = char_string.count("E")

        # increment count
        counter += 1
    # test string at end
    if C_count >= 1 and O_count >= 1 and F_count >= 2 and E_count >= 2:
        chars_in_eight +=1
    #print(char_string)
    trials += 1

print(chars_in_eight/trial_total)
#print(char_string.count("C"))
```

Listing 19: Python example

```
import random

char_list = ["C", "O", "F", "E"]



C_count = 0
O_count = 0
F_count = 0
E_count = 0

char_string = ""
while True:
    my_char = random.choice(char_list)
    char_string += my_char
    # count number of C, O,
    C_count = char_string.count("C")
    O_count = char_string.count("O")
    F_count = char_string.count("F")
    E_count = char_string.count("E")
    if C_count >= 1 and O_count >= 1 and F_count >= 2 and E_count >= 2:
        break
    # increment count


print(char_string)
print( len(char_string) )

#print(char_string.count("C"))
```

Listing 20: Python example

```
import random
```

```
2
3 char_list = ["C", "O", "F", "E"]
4
5
6
7
8 #----------------------------
9 def make_coffee():
10     C_count = 0
11     O_count = 0
12     F_count = 0
13     E_count = 0
14
15     char_string = ""
16     while True:
17         my_char = random.choice(char_list)
18         char_string += my_char
19         # count number of C, O,
20         C_count = char_string.count("C")
21         O_count = char_string.count("O")
22         F_count = char_string.count("F")
23         E_count = char_string.count("E")
24         if C_count >= 1 and O_count >= 1 and F_count >= 2 and E_count >= 2:
25             break
26         # increment count
27
28
29     #print(char_string)
30     #print( len(char_string) )
31     return len(char_string)
32 #--------------------------------
33
34
35 total_trials = 100
36 trials = 1
37
38 total_string_lengths = 0
39
40 while trials <= total_trials:
41     my_number = make_coffee()
42     total_string_lengths += my_number
43     #print(my_number)
44     trials +=1
45 #print(total_string_lengths)
46 average = round(total_string_lengths / total_trials)
```

```
print("On average you will need {} visits".format(average))
```

Listing 21: Python example

```
import random

char_list = ["C", "O", "F", "E"]

string_list = []


def run_trial():
    counter  = 1
    C_count = 0
    O_count = 0
    F_count = 0
    E_count = 0
    char_string = ""
    while True:
        my_char = random.choice(char_list)
        char_string += my_char
        # count number of C, O,
        C_count = char_string.count("C")
        O_count = char_string.count("O")
        F_count = char_string.count("F")
        E_count = char_string.count("E")
        if C_count >= 1 and O_count >= 1 and F_count >= 2 and E_count >= 2:
            string_list.append(char_string)
            break
        # increment count
        counter += 1
    #print(char_string)
    return len(char_string)




total_trials = 1000000
trials = 1
total_chars = 0
while trials <= total_trials:
    total_chars += run_trial()
    trials += 1


print(total_chars)
print(total_trials)
my_average = total_chars/total_trials
```

```
print(my_average)

#print(string_list)
string_list.sort(key = len)
#print(string_list)
print(string_list[0])
print(string_list[-1])
print(len(string_list[0]))
print(len(string_list[-1]))

my_dict = {}
for i in range(len(string_list[0]), len(string_list[-1])+1 ):
    my_dict[i]=0
    #print(i)

#print(my_dict)
for x in string_list:
    my_dict[len(x)]+=1

print(my_dict)
my_sum = 0
for x in my_dict:
    my_sum += my_dict[x]
print(my_sum)
# dictionary not the same lenght as string list
for x,y in my_dict.items():
    temp = int(y)
    temp = temp/total_trials
    my_dict[x]= temp
print(my_dict)

my_file = open("distribution_next.txt","a")
for x,y in my_dict.items():
    my_string = "{},{:.6f}\n".format(x,y)
    my_file.write(my_string)

#my_file.write("hello")
#my_file.write("\n")
#my_file.write("goodbye")
my_file.close()
#my_file = open("MyFile.txt","w")
#my_file.write("")
#my_file.close()
```

Listing 22: Python example

12 Tests

```
# roll a dice
import random
two_d_list = []



def searchfor(n):
    if len(two_d_list)== 0:
        two_d_list.append([n,1])
        return
    added = False
    for sublist in two_d_list:
        if sublist[0]==n:
            sublist[1] += 1
            added = True
            break
    if added == False:
        two_d_list.append([n,1])

for i in range(0,1500):
    diceroll = random.randint(1,6)+random.randint(1,6)+random.randint(1,6)
    searchfor(diceroll)
    #print(diceroll)


two_d_list.sort()
print(two_d_list)
```

Listing 23: Dice Roll

```
guess_list = [45,78,90,21,7]

guess=int(input("Enter your guess"))


while guess in guess_list:
    guess = int(input("You have already tried this number,\
have another guess"))
guess_list.append(guess)
print(guess_list)
```

Listing 24: Python example

```
import time
start= time.time()
t= time.localtime()
```

```
print(t)
print(start)



word = "halloween"
char_list = list(word)
print(char_list)
char_list_chosen=["l", "w", "e"]
outstring = ""
for x in char_list:
    if x in char_list_chosen:
        outstring += " "+x+" "
    else:
        outstring += " _ "
print(outstring)




question_one = {
    "question": "What is the capital of France?",
    "A": "Paris",
    "B": "Moscow",
    "C": "Shanghai",
    "answer": "C"
    }

print(question_one["question"])
for x,y in question_one.items():
    if x != "question" and x != "answer":
        print("{}) {}".format(x,y))

for x,y in question_one.items():
    if len(x)== 1:
        print("{}) {}".format(x,y))

answer = input("Enter your choice ")
if answer == question_one["answer"]:
    print(" Brilliant ")
end=time.time()
print(end)
print(end-start)
count = 0
```

```
while count < 3:
    temp={}
    temp["english"]= input("Enter your english word")
    temp["thai"] = input("Enter your thai word")
    print(temp)
    count +=1
```

Listing 25: Python example

```
my_num=0.0000067
print("{:.6f}".format(my_num) )
```

Listing 26: Python example

```
word = "halloween"

def pic(c):
    if c >= 1:
        print(" "*30+"    _ _ _ _")
    if c >=2:
        print(" "*30+"    |     |")
    if c >=3:
        print(" "*30+"   ( )    |")
    if c >= 4:
        print(" "*30+"    |     |")
    if c >= 5:
        print(" "*30+"   ---    |")
        print(" "*30+"    |     |")
        print(" "*30+"   / \    |")
        print(" "*30+"        __|")
pic(5)
char_list = list(word)
char_individual_list=[]
char_list_chosen=[]

for x in char_list:
    if x not in char_individual_list:
        char_individual_list.append(x)

wrong_count = 0
while len(char_individual_list)>0:
    outstring = ""
    for x in char_list:
        if x in char_list_chosen:
            outstring += " "+x+" "
        else:
            outstring += " _ "
    print("{}\n\n".format(outstring))
```

```
    char = input("guess a letter: ")
    if char in char_individual_list:
        char_list_chosen.append(char)
        char_individual_list.remove(char)
    else:
        wrong_count += 1
        pic(wrong_count)
    if len(char_individual_list) == 0:
        print("you got it")


outstring = ""
for x in char_list:
    if x in char_list_chosen:
        outstring += " "+x+" "
    else:
        outstring += " _ "
print(outstring)
```

Listing 27: Python example

13 Strings

```
my_string="adkadkkdjk  jkjdhkdhdh"
my_split=my_string.split("a")
print(my_split)
my_list=list(my_string)
print(my_list)
```

Listing 28: Python example

```
my_string = "woolloomooloo"

my_letters=[]
count_letters = []

for x in my_string:
    if x not in my_letters:
        my_letters.append(x)
        count_letters.append(my_string.count(x))

print(my_letters)
print(count_letters)
```

Listing 29: Python example

14 Objects

```
class Shark:

    def __init__(self, name, age):
        self.name = name
        self.age = age
        """ constructor """
        print("This is the constructor")

    def swim(self):
        """ what the shark is doing"""
        print(self.name+" is swimming")

    def be_awesome(self):
        """ what the shark is being"""
        print(self.name+" is being awesome")

    def _age(self):
        """ how old is it """
        print(self.name +" is "+str(self.age)+" years old")


def main():
    sammy = Shark("Sammy", 5)
    stevie = Shark("Stevie", 3)
    sammy.swim()
    stevie.be_awesome()
    sammy._age()

if __name__ =="__main__":
    main()
```

Listing 30: Python example

```
class Store:

    def __init__(self):
        self.colour = "rgb(0,0,0,0)"









class Shark:

    def __init__(self, name, age):
        self.name = name
        self.age = age
        """ constructor """
```

```
        store.colour = "rgb(255,255,255,1)"
        print("This is the constructor")

    def swim(self):
        """ what the shark is doing"""
        print(self.name+" is swimming")

    def be_awesome(self):
        """ what the shark is being"""
        print(self.name+" is being awesome")

    def _age(self):
        """ how old is it """
        print(self.name +" is "+str(self.age)+" years old")

store = Store()
shark = Shark("sam", 10)
print(store.colour)
```

Listing 31: Python example

```
class Question:

    def __init__(self, question,answer, score):

        self.question = question
        self.answer = answer
        self.score = score

        self.user_answer = ""
        self.user_score = 0
        self.response_to_user = ""

    def make_question(self):
        msg = self.question
        self.user_answer = input(msg)
        if self.user_answer == self.answer:
            self.response_to_user = "Correct"
            self.user_score = self.score
        else:
            self.response_to_user = "Incorrect"

    def feedback(self):
        strOne = "For the question: "+self.question+ "\n"
        strTwo = "You answered: "+self.user_answer +"\n"
        strThree = "Your answer was "+self.response_to_user
        print(strOne + strTwo + strThree)

```

```
#q_one = Question("What is the capital of France?", "Paris", 10)
#q_one.make_question()
#q_one.feedback()
q_list = []
q_list.append(Question("What is the capital of France?", "Paris", 10))
q_list.append(Question("What is the capital of Belgium?", "Brussels", 10))
q_list.append(Question("What is the capital of Holland?", "Amstredam", 10))

count = 0
while count < len(q_list):
    q_list[count].make_question()
    count += 1

count = 0
while count < len(q_list):
    q_list[count].feedback()
    count += 1
```

Listing 32: Python example