Part 1

1. Evaluate the following expressions for `num1 = 10` and `num2 = 20`.

(a) `not (num1 < 1) and num2 < 10`

`Ans: The above given expressions is false.`

(b) `not (num1 < 1) and num2 < 10 or num1 + num3 < 100`

`Ans: Gives error as num3 is not defined.`

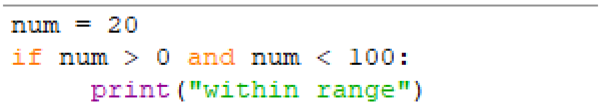
(c) `not (num2 > 1) or num1 > num2 - 10`

`Ans: The above expression is false.`

2. Give an appropriate if statement for each of the following (the value of `num` is not important):

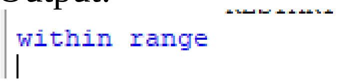
(a) Displays `'within range'` if `num` is between 0 and 100, inclusive.

Code:



```
num = 20
if num > 0 and num < 100:
     print("within range")
```

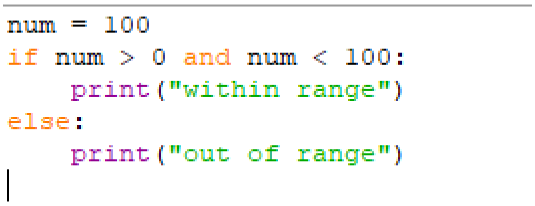
Output:



```
within range
```

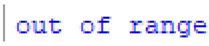
(b) Displays `'within range'` if `num` is between 0 and 100, inclusive, and displays `'out of range'` otherwise.

Code:



```
num = 100
if num > 0 and num < 100:
    print("within range")
else:
    print("out of range")
```

Output

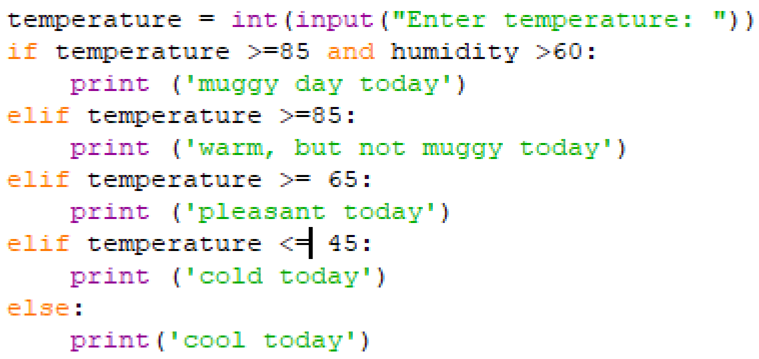


```
out of range
```

3. Rewrite the following if-else statements using a single if statement and elif:

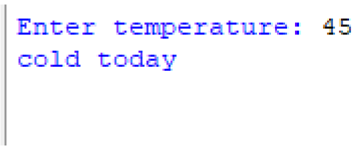
```
if temperature >= 85 and humidity > 60:
    print ('muggy day today')
else:
    if temperature >= 85:
         print ('warm, but not muggy today')
     else:
        if temperature >= 65:
             print ('pleasant today')
        else:
             if temperature <= 45:
                 print ('cold today')
            else:
                 print ('cool today')
```

Code:



```
temperature = int(input("Enter temperature: "))
if temperature >=85 and humidity >60:
    print ('muggy day today')
elif temperature >=85:
    print ('warm, but not muggy today')
elif temperature >= 65:
    print ('pleasant today')
elif temperature <= 45:
    print ('cold today')
else:
    print('cool today')
```

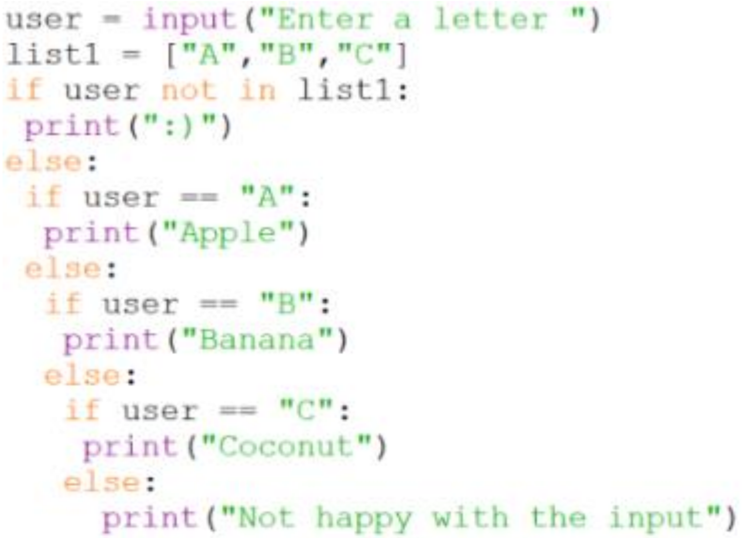
Output:



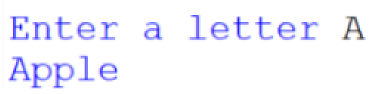
```
Enter temperature: 45
cold today
```

4. Write a Python program in which:

(a) The user enters either `'A', 'B', or 'C'`. If `'A'` is entered, the program should display the word `'Apple'`; if `'B'` is entered, it displays `'Banana'`; and if 'C' is entered, it displays `'Coconut'`. Use nested if statements for this.

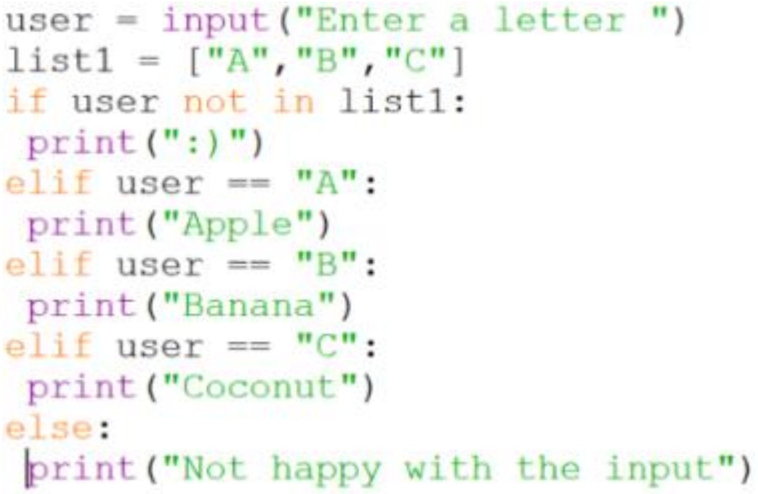


```
user = input("Enter a letter ")
list1 = ["A","B","C"]
if user not in list1:
 print(":)")
else:
 if user == "A":
  print("Apple")
 else:
  if user == "B":
   print("Banana")
  else:
   if user == "C":
    print("Coconut")
   else:
     print("Not happy with the input")
```

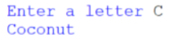


```
Enter a letter A
Apple
```

(b) Repeat question **(a)** using an if statement with `elif` headers instead.

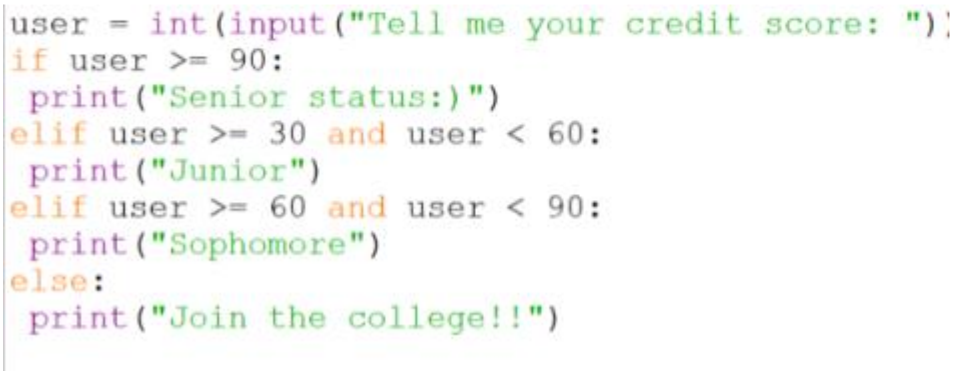


```
user = input("Enter a letter ")
list1 = ["A","B","C"]
if user not in list1:
 print(":)")
elif user == "A":
 print("Apple")
elif user == "B":
 print("Banana")
elif user == "C":
 print("Coconut")
else:
 print("Not happy with the input")
```

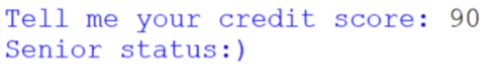


```
Enter a letter C
Coconut
```

(c) A student enters the number of college credits earned. If the number of credits is greater than or equal to 90, 'Senior Status' is displayed; if greater than or equal to 60, 'Junior Status' is displayed; if greater than or equal to 30, 'Sophomore Status' is displayed; else, 'Freshman Status' is displayed.

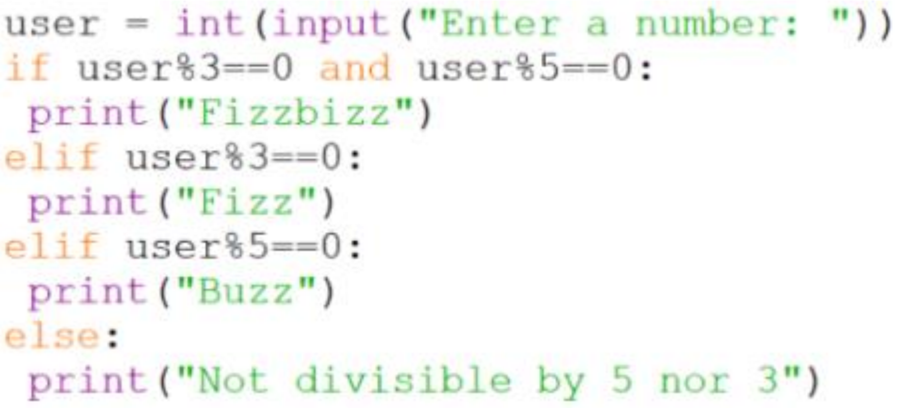


```
user = int(input("Tell me your credit score: "))
if user >= 90:
 print("Senior status:)")
elif user >= 30 and user < 60:
 print("Junior")
elif user >= 60 and user < 90:
 print("Sophomore")
else:
 print("Join the college!!")
```

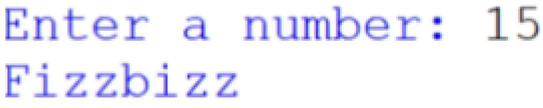


```
Tell me your credit score: 90
Senior status:)
```

(d) The user enters a number. If the number is divisible by 3, the word ‘Fizz’ should be displayed; if the number is divisible by 5 the word ‘Buzz’ should be displayed and if the number is divisible by both ‘FizzBuzz’ should be displayed.

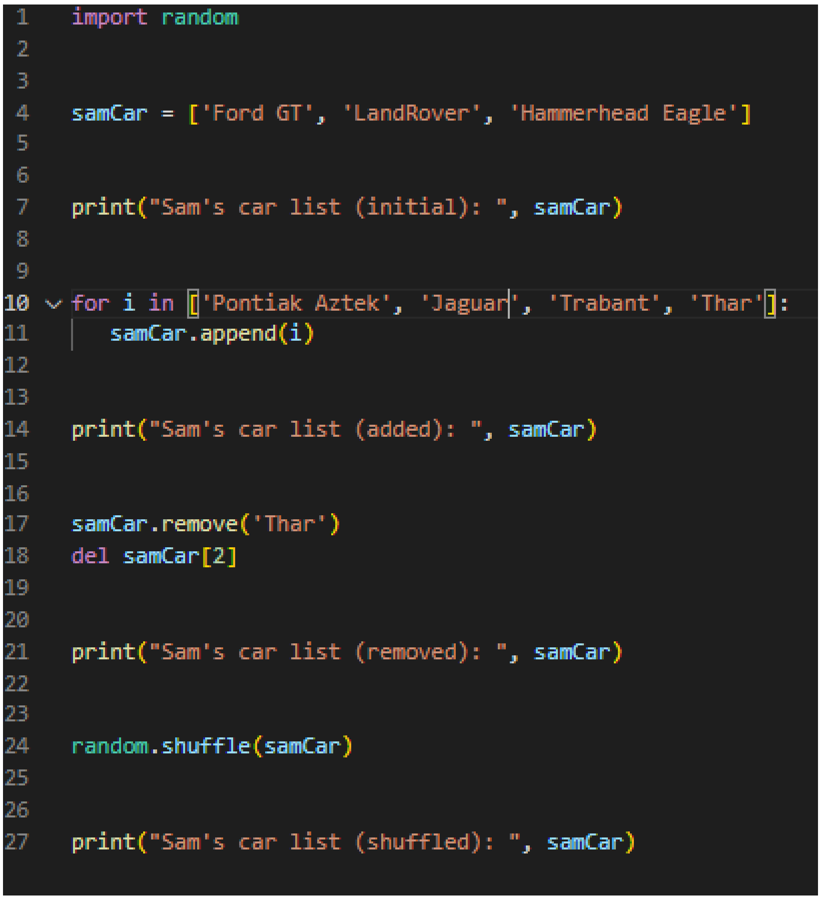


```
user = int(input("Enter a number: "))
if user%3==0 and user%5==0:
 print("Fizzbizz")
elif user%3==0:
 print("Fizz")
elif user%5==0:
 print("Buzz")
else:
 print("Not divisible by 5 nor 3")
```



```
Enter a number: 15
Fizzbizz
```

5. Sam wants to store his series of car to a list. The list of a car are: (up to you). After creating a list he add some car and delete some car and at last there are still 5 cars left in his list. Additionally, he wants his car to be shuffled every time when the list is being displayed. [Hint: shuffle from random]

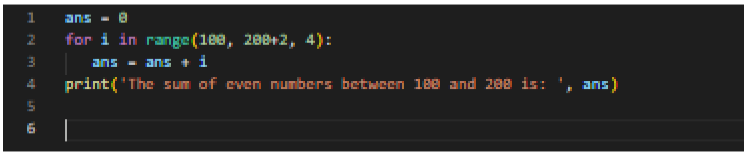


```python
import random


samCar = ['Ford GT', 'LandRover', 'Hammerhead Eagle']


print("Sam's car list (initial): ", samCar)


for i in ['Pontiak Aztek', 'Jaguar', 'Trabant', 'Thar']:
    samCar.append(i)


print("Sam's car list (added): ", samCar)


samCar.remove('Thar')
del samCar[2]


print("Sam's car list (removed): ", samCar)


random.shuffle(samCar)


print("Sam's car list (shuffled): ", samCar)
```

Part 2

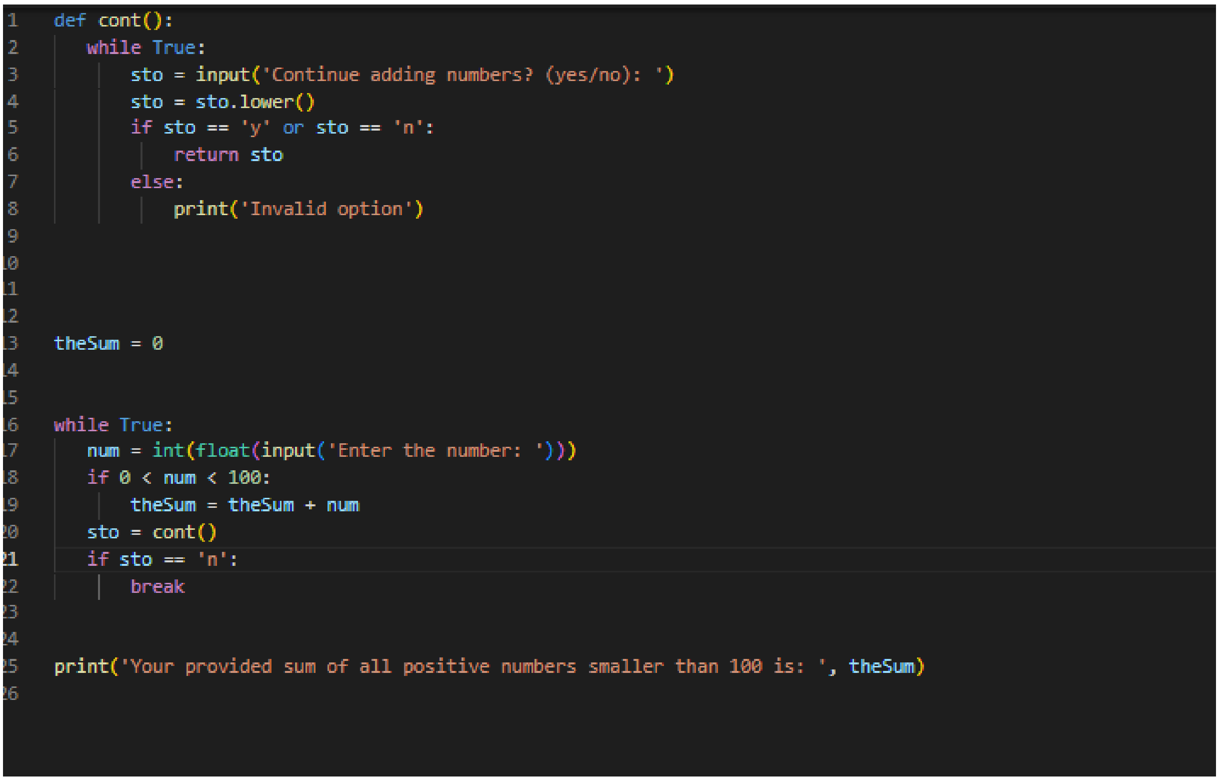
1. Write a program that:

(a) Uses a loop to add up all the even numbers between 100 and 200, inclusive.



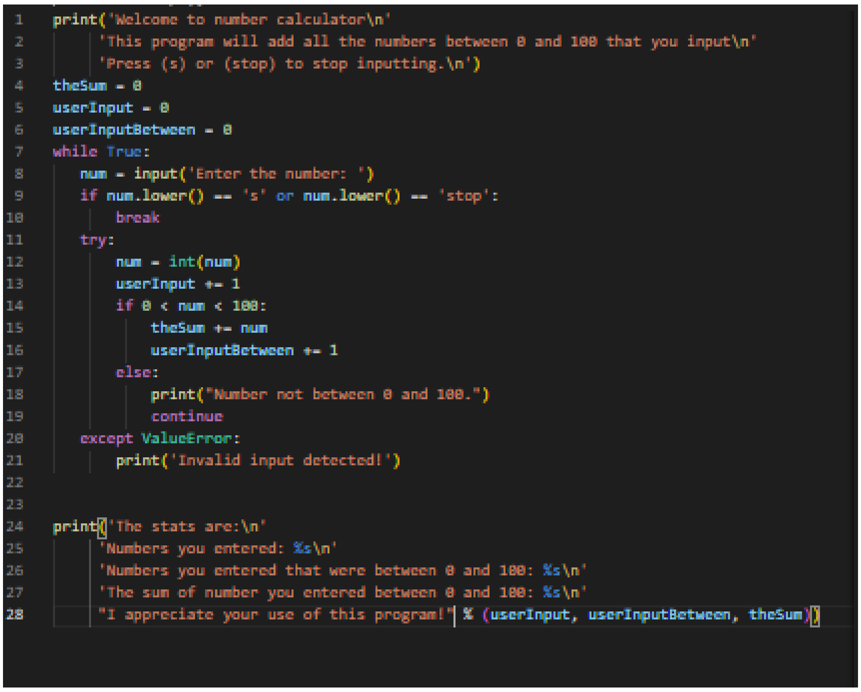
```python
ans = 0
for i in range(100, 200+2, 4):
    ans = ans + i
print('The sum of even numbers between 100 and 200 is: ', ans)

```

(b) Sums a series of (positive) integers entered by the user, excluding all numbers that are greater than 100.



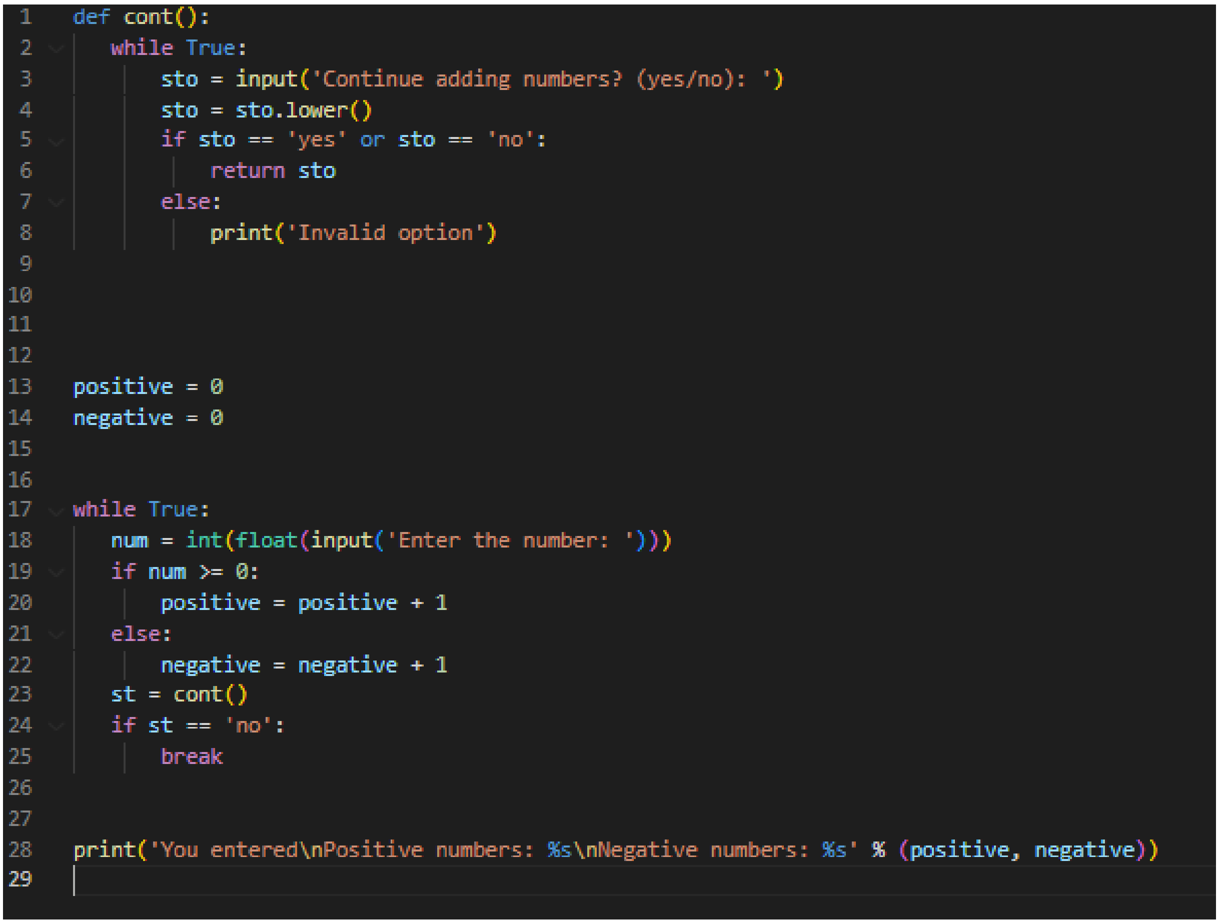
```python
def cont():
    while True:
        sto = input('Continue adding numbers? (yes/no): ')
        sto = sto.lower()
        if sto == 'y' or sto == 'n':
            return sto
        else:
            print('Invalid option')




theSum = 0


while True:
    num = int(float(input('Enter the number: ')))
    if 0 < num < 100:
        theSum = theSum + num
    sto = cont()
    if sto == 'n':
        break


print('Your provided sum of all positive numbers smaller than 100 is: ', theSum)

```

(c) Solves Q2 but this time using an infinite loop, break and continue statements.



```python
print('Welcome to number calculator\n'
      'This program will add all the numbers between 0 and 100 that you input\n'
      'Press (s) or (stop) to stop inputting.\n')
theSum = 0
userInput = 0
userInputBetween = 0
while True:
    num = input('Enter the number: ')
    if num.lower() == 's' or num.lower() == 'stop':
        break
    try:
        num = int(num)
        userInput += 1
        if 0 < num < 100:
            theSum += num
            userInputBetween += 1
        else:
            print("Number not between 0 and 100.")
            continue
    except ValueError:
        print('Invalid input detected!')


print('The stats are:\n'
      'Numbers you entered: %s\n'
      'Numbers you entered that were between 0 and 100: %s\n'
      'The sum of number you entered between 0 and 100: %s\n'
      "I appreciate your use of this program!" % (userInput, userInputBetween, theSum))
```

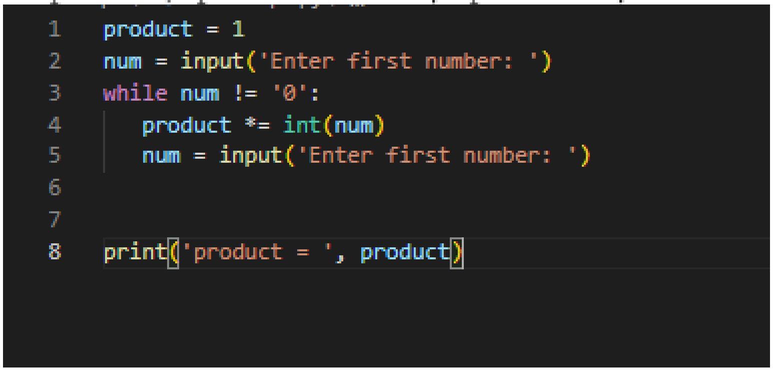
(d) Prompts the user to enter any number of positive and negative integer values, then displays the number of each type that were entered.



```python
def cont():
    while True:
        sto = input('Continue adding numbers? (yes/no): ')
        sto = sto.lower()
        if sto == 'yes' or sto == 'no':
            return sto
        else:
            print('Invalid option')




positive = 0
negative = 0


while True:
    num = int(float(input('Enter the number: ')))
    if num >= 0:
        positive = positive + 1
    else:
        negative = negative + 1
    st = cont()
    if st == 'no':
        break


print('You entered\nPositive numbers: %s\nNegative numbers: %s' % (positive, negative))
```

2. The following while loop is meant to multiply a series of integers input by the user, until a sentinel value of 0 is entered. Indicate any errors in the code given. See if you can fix the program and get it running.

```
product = 1
num = input('Enter first number: ')
while num != 0:
    num = input('Enter first number: )
    product = product * num
  print('product = ', product)
```

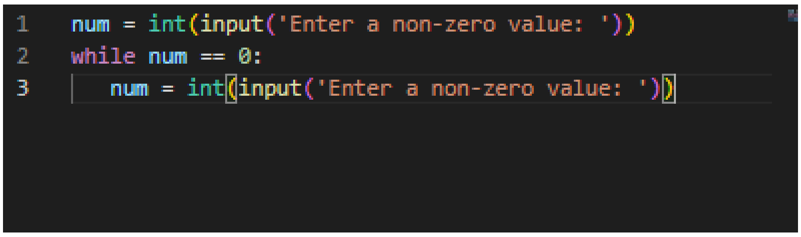


```
product = 1
num = input('Enter first number: ')
while num != '0':
    product *= int(num)
    num = input('Enter first number: ')


print('product = ', product)
```

3. For each of the following, indicate which the definite loop is, and which an indefinite loop, explain your reasoning.

(a)

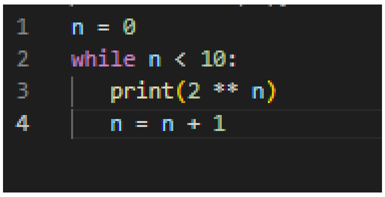
```
num = input('Enter a non-zero value:')
while num == 0:
    num = input('Enter a non-zero value: ')
```



```
num = int(input('Enter a non-zero value: '))
while num == 0:
    num = int(input('Enter a non-zero value: '))
```

(b)

```
num = 0
while n < 10:
    print 2 ** n
    n = n + 1
```



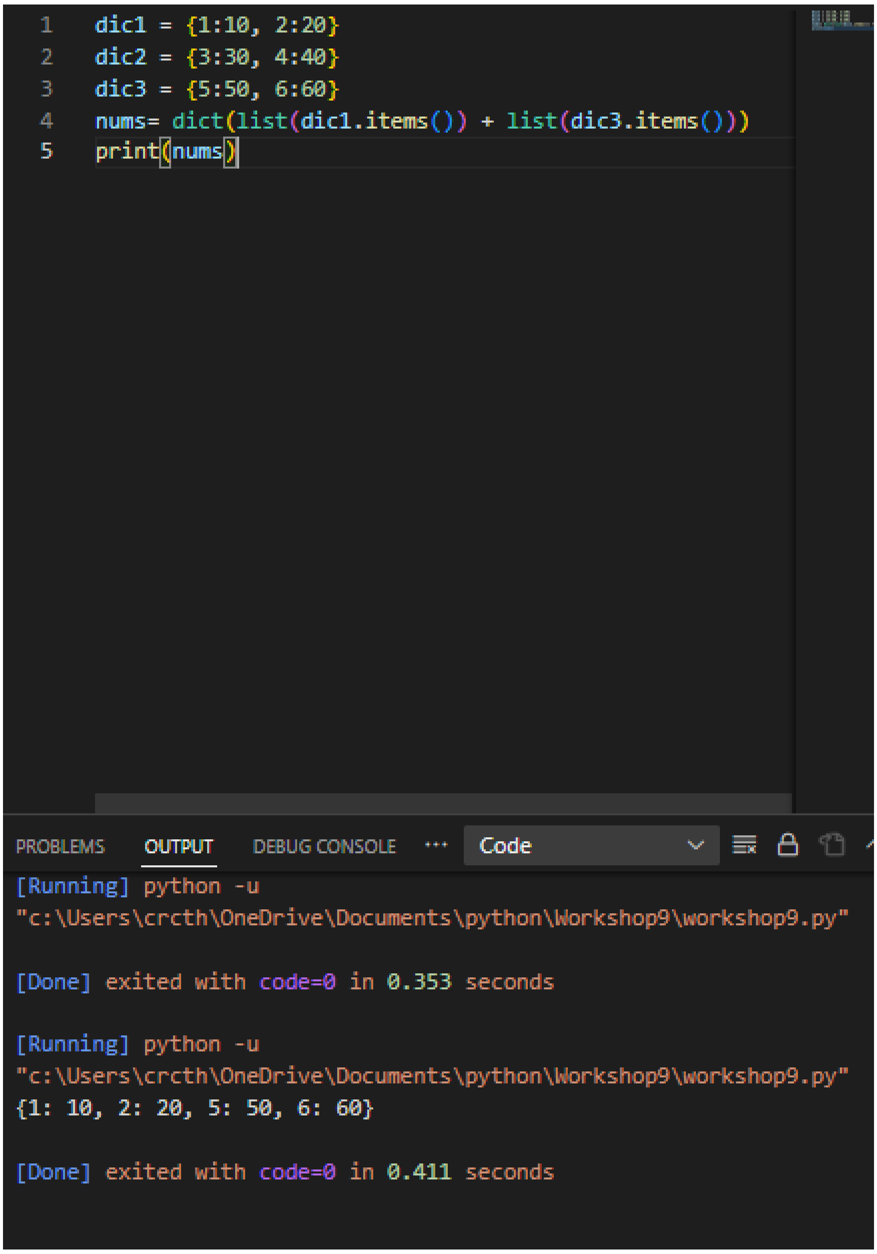
```
n = 0
while n < 10:
    print(2 ** n)
    n = n + 1
```

Part 3

1. Create three dictionaries:

```
dic1 = {1:10, 2:20}
dic2 = {3:30, 4:40}
dic3 = {5:50, 6:60}
```

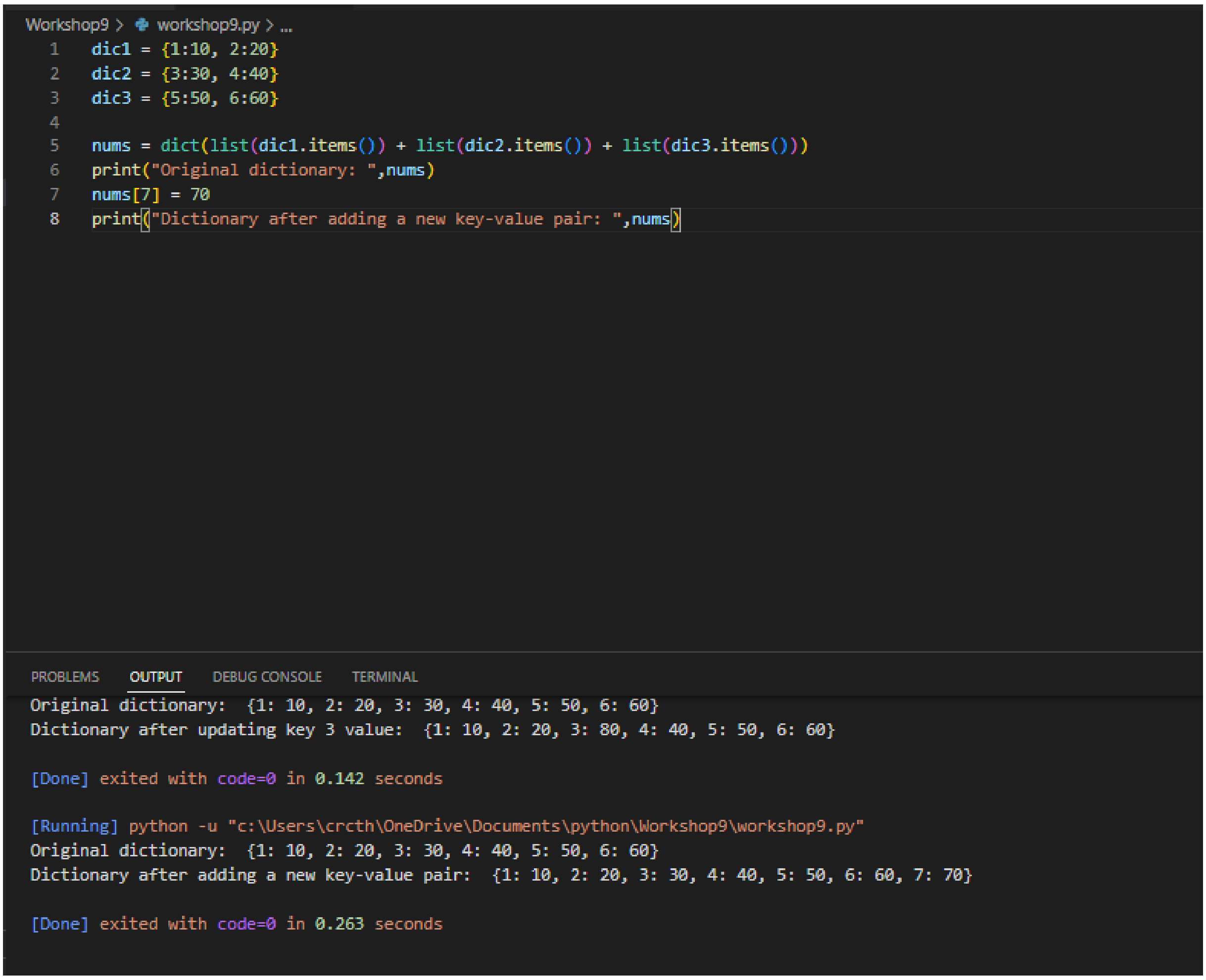
(a) Write code to concatenate these dictionaries to create a new one. Create a variable called `nums` to store the resulting dictionary. There are multiple ways to do this, however, one of the easiest is to convert each of the dictionaries items to a list (which can be added together) and pass them to the `dict()` constructor.



```
dic1 = {1:10, 2:20}
dic2 = {3:30, 4:40}
dic3 = {5:50, 6:60}
nums= dict(list(dic1.items()) + list(dic3.items()))
print(nums)
```

```
[Running] python -u
"c:\Users\crcth\OneDrive\Documents\python\Workshop9\workshop9.py"

[Done] exited with code=0 in 0.353 seconds

[Running] python -u
"c:\Users\crcth\OneDrive\Documents\python\Workshop9\workshop9.py"
{1: 10, 2: 20, 5: 50, 6: 60}

[Done] exited with code=0 in 0.411 seconds
```

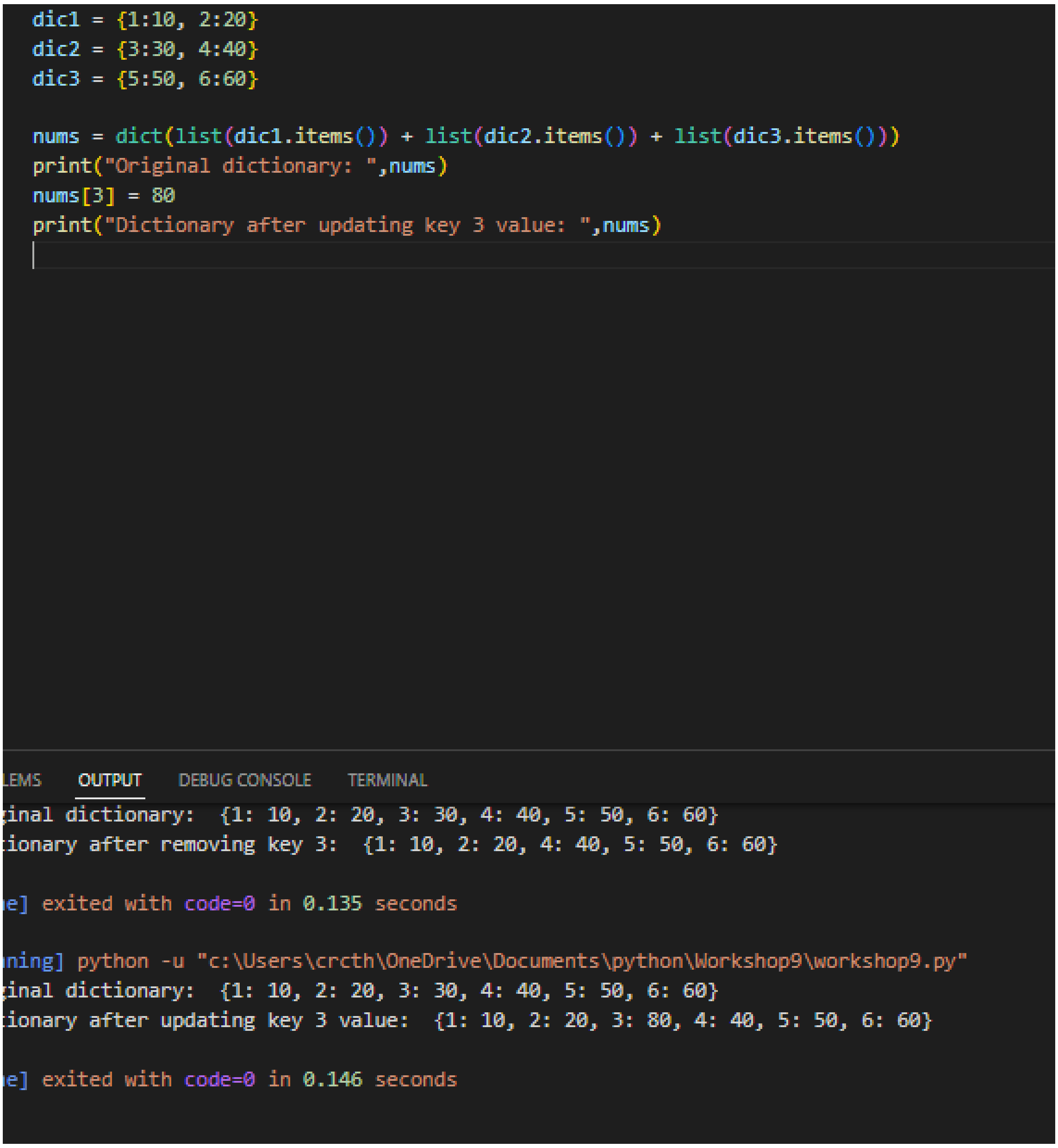
(b) Write code to add a new key/value pair to the dictionary `nums`**:** `(7, 70)`



```python
dic1 = {1:10, 2:20}
dic2 = {3:30, 4:40}
dic3 = {5:50, 6:60}

nums = dict(list(dic1.items()) + list(dic2.items()) + list(dic3.items()))
print("Original dictionary: ",nums)
nums[7] = 70
print("Dictionary after adding a new key-value pair: ",nums)
```

```
Original dictionary:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
Dictionary after updating key 3 value:  {1: 10, 2: 20, 3: 80, 4: 40, 5: 50, 6: 60}

[Done] exited with code=0 in 0.142 seconds

[Running] python -u "c:\Users\crcth\OneDrive\Documents\python\Workshop9\workshop9.py"
Original dictionary:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
Dictionary after adding a new key-value pair:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60, 7: 70}

[Done] exited with code=0 in 0.263 seconds
```

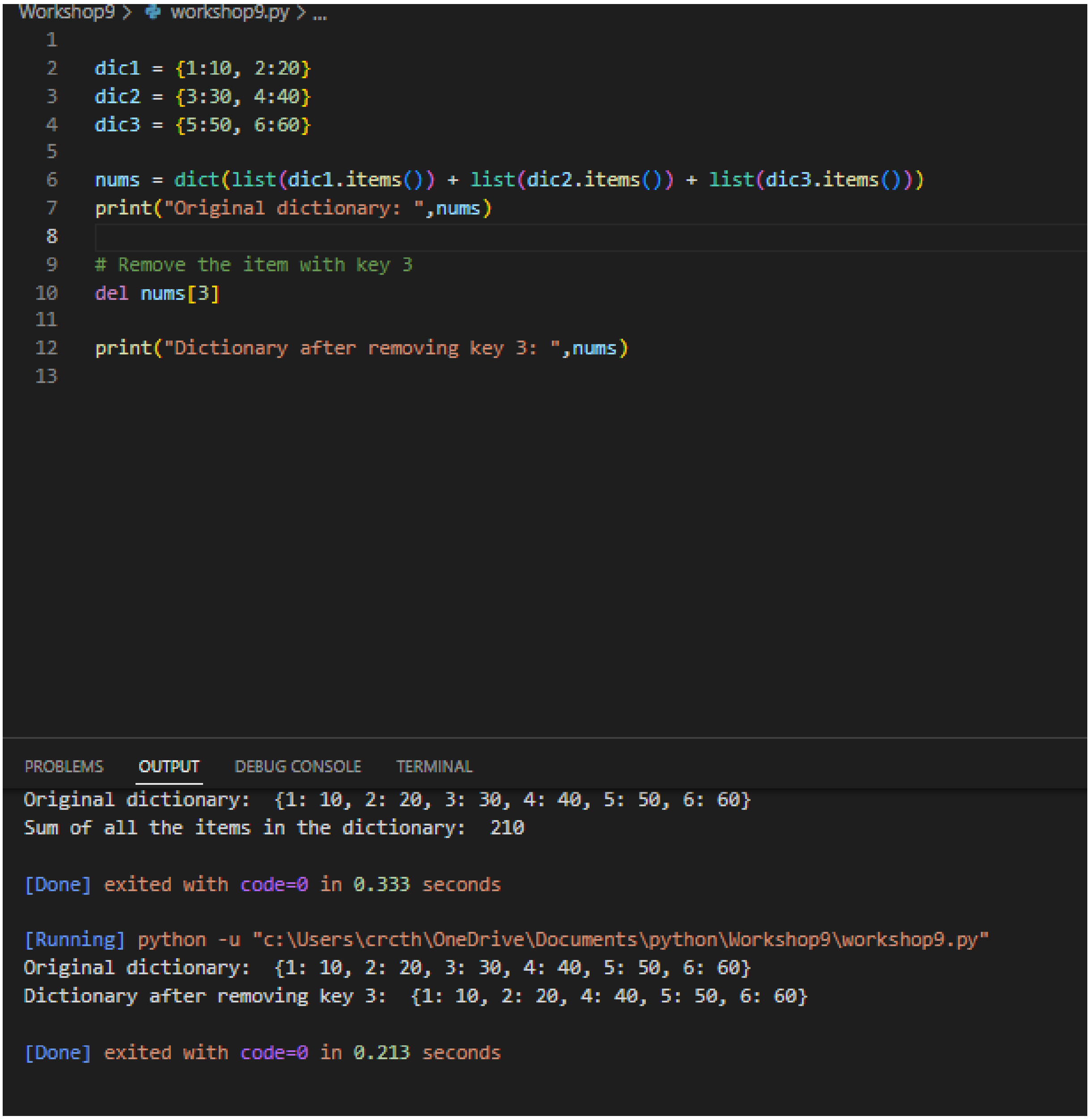
(c) Write code to update the value of the item with key `3` in `nums` to `80`



```
dic1 = {1:10, 2:20}
dic2 = {3:30, 4:40}
dic3 = {5:50, 6:60}

nums = dict(list(dic1.items()) + list(dic2.items()) + list(dic3.items()))
print("Original dictionary: ",nums)
nums[3] = 80
print("Dictionary after updating key 3 value: ",nums)
```

```
inal dictionary:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
ionary after removing key 3:  {1: 10, 2: 20, 4: 40, 5: 50, 6: 60}

e] exited with code=0 in 0.135 seconds

ning] python -u "c:\Users\crcth\OneDrive\Documents\python\Workshop9\workshop9.py"
inal dictionary:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
ionary after updating key 3 value:  {1: 10, 2: 20, 3: 80, 4: 40, 5: 50, 6: 60}

e] exited with code=0 in 0.146 seconds
```

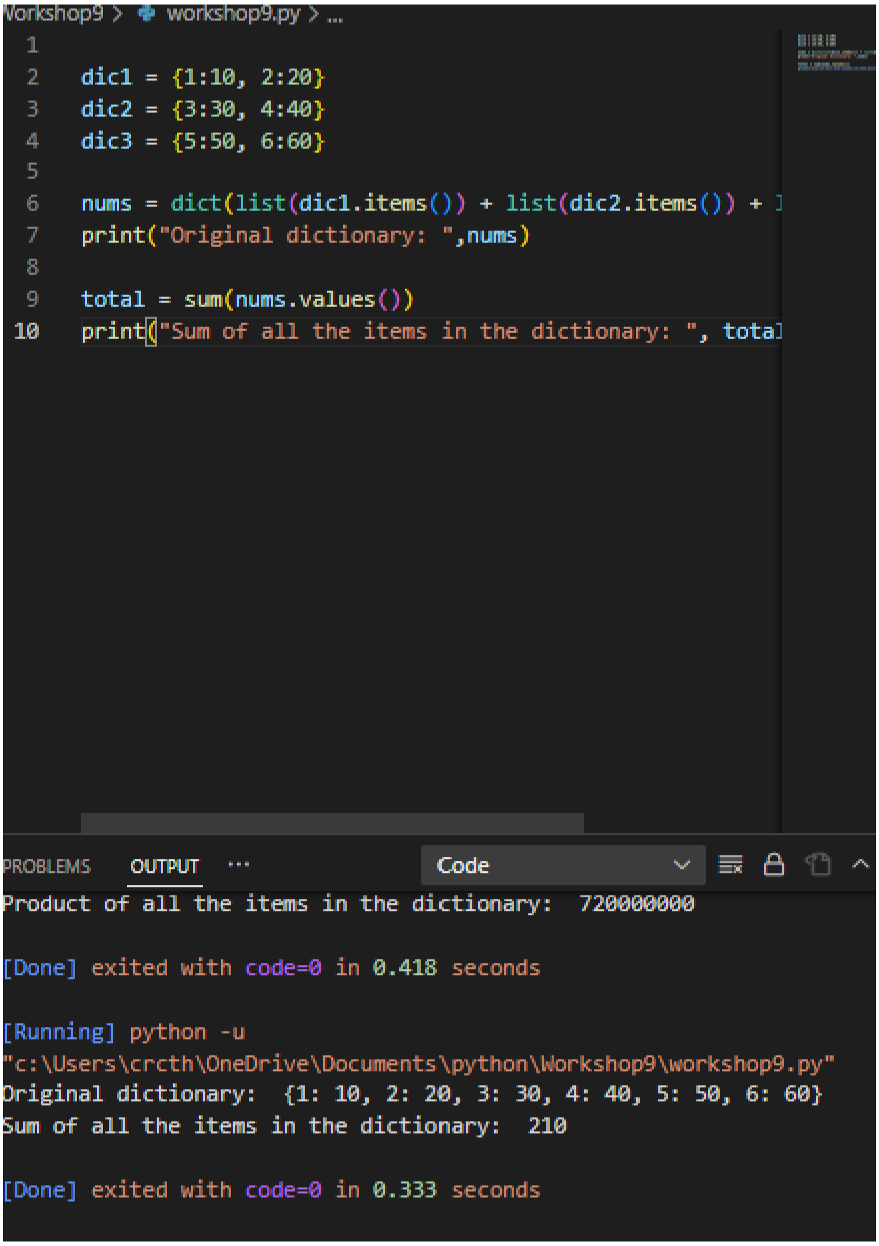
(d) Write code to remove the third item from dictionary `nums`.



```python

dic1 = {1:10, 2:20}
dic2 = {3:30, 4:40}
dic3 = {5:50, 6:60}

nums = dict(list(dic1.items()) + list(dic2.items()) + list(dic3.items()))
print("Original dictionary: ",nums)

# Remove the item with key 3
del nums[3]

print("Dictionary after removing key 3: ",nums)

```

```
Original dictionary:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
Sum of all the items in the dictionary:  210

[Done] exited with code=0 in 0.333 seconds

[Running] python -u "c:\Users\crcth\OneDrive\Documents\python\Workshop9\workshop9.py"
Original dictionary:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
Dictionary after removing key 3:  {1: 10, 2: 20, 4: 40, 5: 50, 6: 60}

[Done] exited with code=0 in 0.213 seconds
```

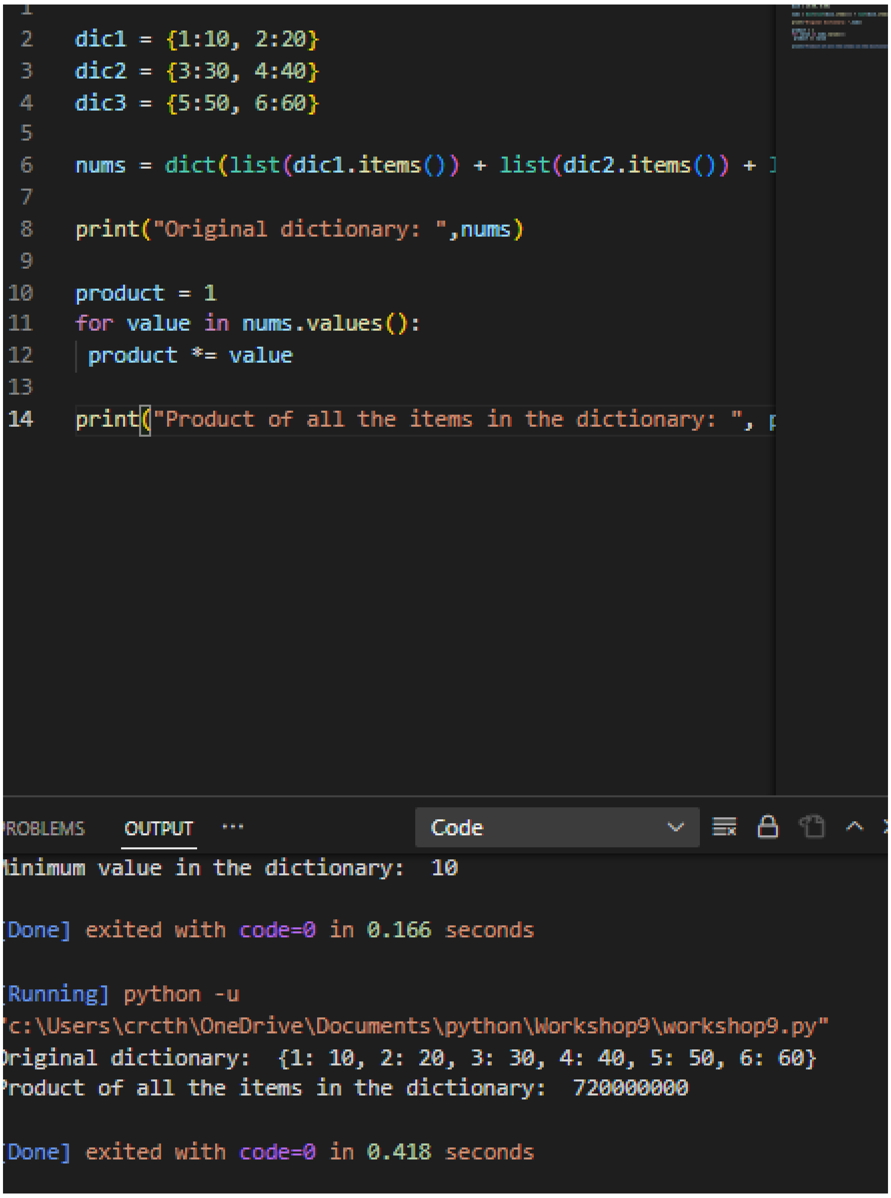
(e) Write code to sum all the items in the dictionary `nums`



```python

dic1 = {1:10, 2:20}
dic2 = {3:30, 4:40}
dic3 = {5:50, 6:60}

nums = dict(list(dic1.items()) + list(dic2.items()) + l
print("Original dictionary: ",nums)

total = sum(nums.values())
print("Sum of all the items in the dictionary: ", total
```

```
Product of all the items in the dictionary:  720000000

[Done] exited with code=0 in 0.418 seconds

[Running] python -u
"c:\Users\crcth\OneDrive\Documents\python\Workshop9\workshop9.py"
Original dictionary:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
Sum of all the items in the dictionary:  210

[Done] exited with code=0 in 0.333 seconds
```

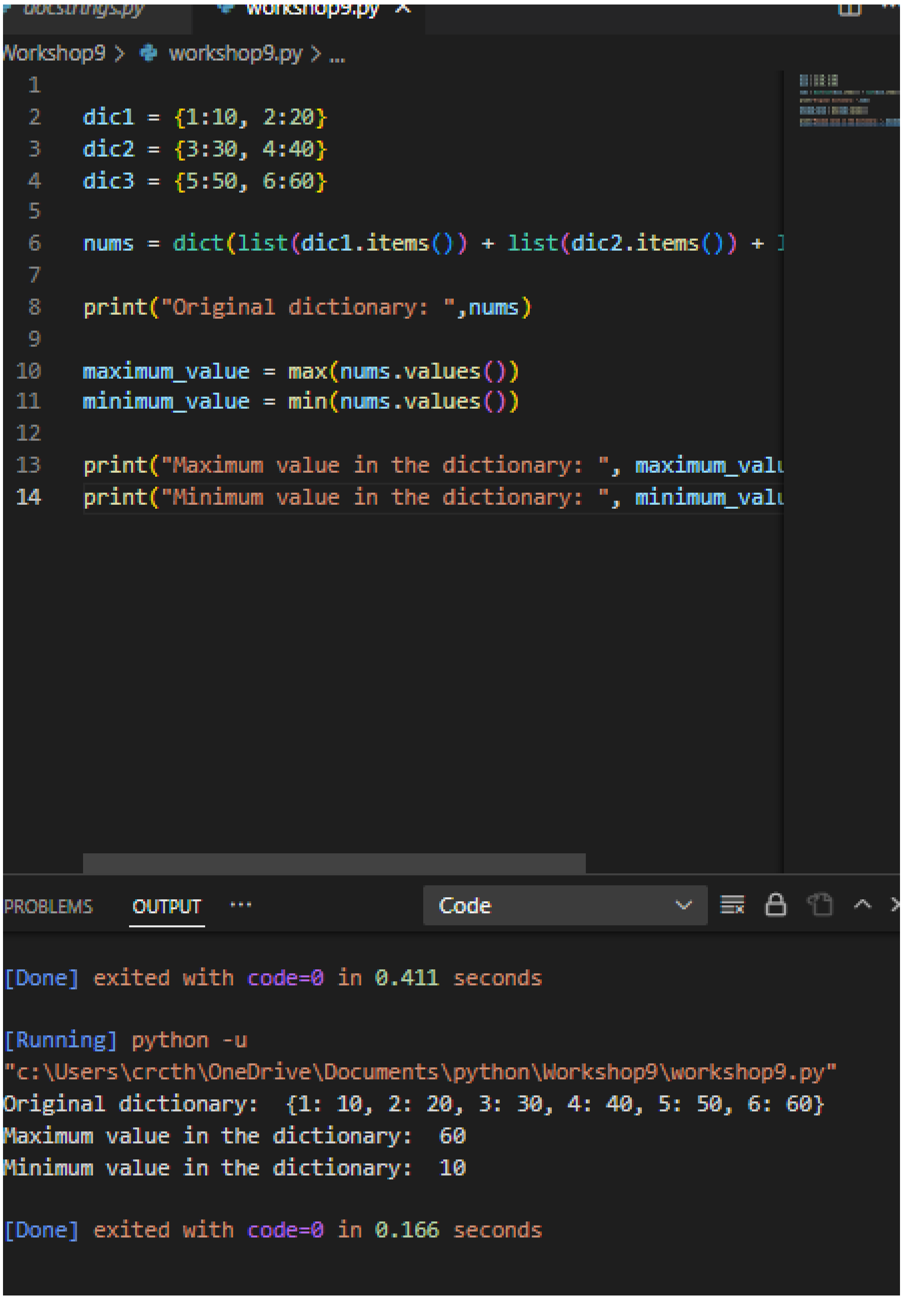
(f) Write code to multiply all the items in the dictionary `nums`



```
dic1 = {1:10, 2:20}
dic2 = {3:30, 4:40}
dic3 = {5:50, 6:60}

nums = dict(list(dic1.items()) + list(dic2.items()) + l

print("Original dictionary: ",nums)

product = 1
for value in nums.values():
    product *= value

print("Product of all the items in the dictionary: ", p
```

```
Minimum value in the dictionary:  10

[Done] exited with code=0 in 0.166 seconds

[Running] python -u
"c:\Users\crcth\OneDrive\Documents\python\Workshop9\workshop9.py"
Original dictionary:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
Product of all the items in the dictionary:  720000000

[Done] exited with code=0 in 0.418 seconds
```

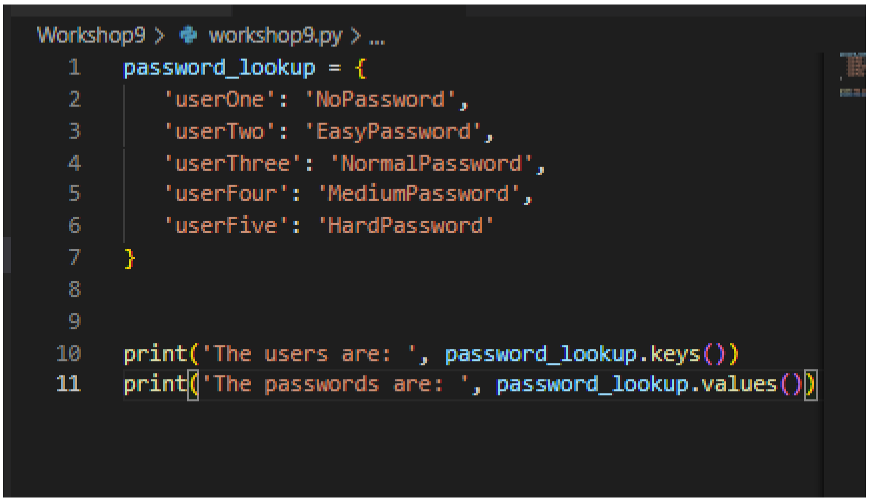
(g) Write code to retrieve the maximum and minimum values in `nums`.



```python

dic1 = {1:10, 2:20}
dic2 = {3:30, 4:40}
dic3 = {5:50, 6:60}

nums = dict(list(dic1.items()) + list(dic2.items()) + l

print("Original dictionary: ",nums)

maximum_value = max(nums.values())
minimum_value = min(nums.values())

print("Maximum value in the dictionary: ", maximum_valu
print("Minimum value in the dictionary: ", minimum_valu
```

```
[Done] exited with code=0 in 0.411 seconds

[Running] python -u
"c:\Users\crcth\OneDrive\Documents\python\Workshop9\workshop9.py"
Original dictionary:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
Maximum value in the dictionary:  60
Minimum value in the dictionary:  10

[Done] exited with code=0 in 0.166 seconds
```

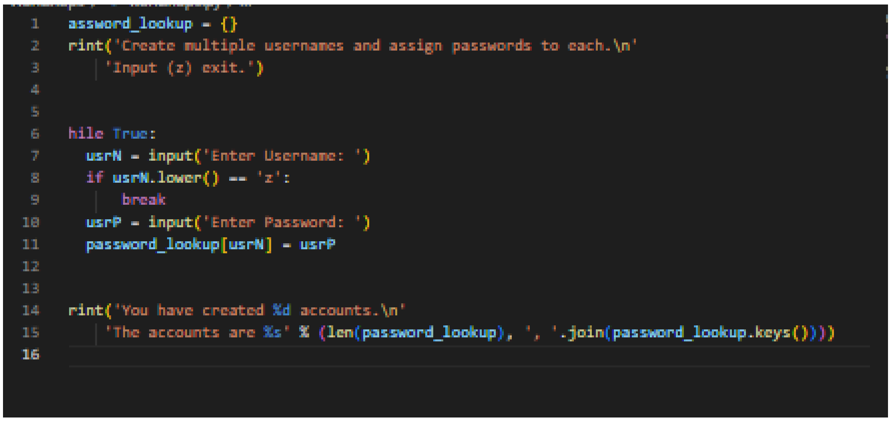
3. Create a dictionary named `password_lookup` that contains usernames as keys and passwords as associated string values. Make up data for five entries.



```python
password_lookup = {
    'userOne': 'NoPassword',
    'userTwo': 'EasyPassword',
    'userThree': 'NormalPassword',
    'userFour': 'MediumPassword',
    'userFive': 'HardPassword'
}


print('The users are: ', password_lookup.keys())
print('The passwords are: ', password_lookup.values())
```

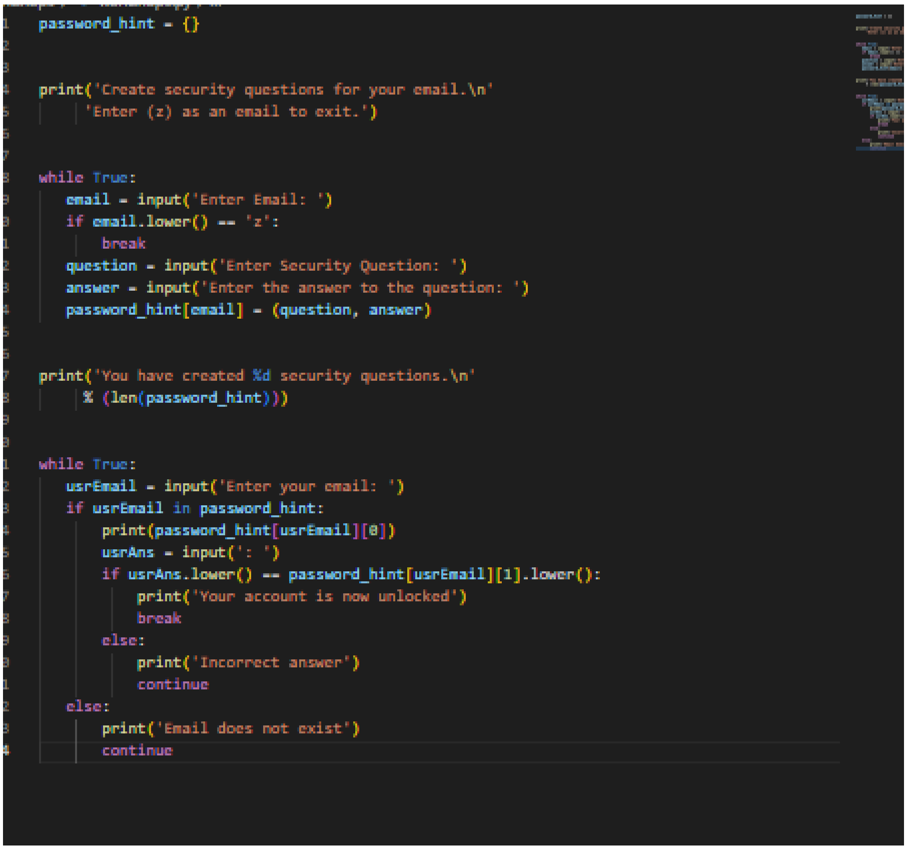
4. Write a program that creates an initially empty dictionary named `password_lookup`, prompting one-by-one for usernames and passwords

(until a username of 'z' is read) entering each into the dictionary.



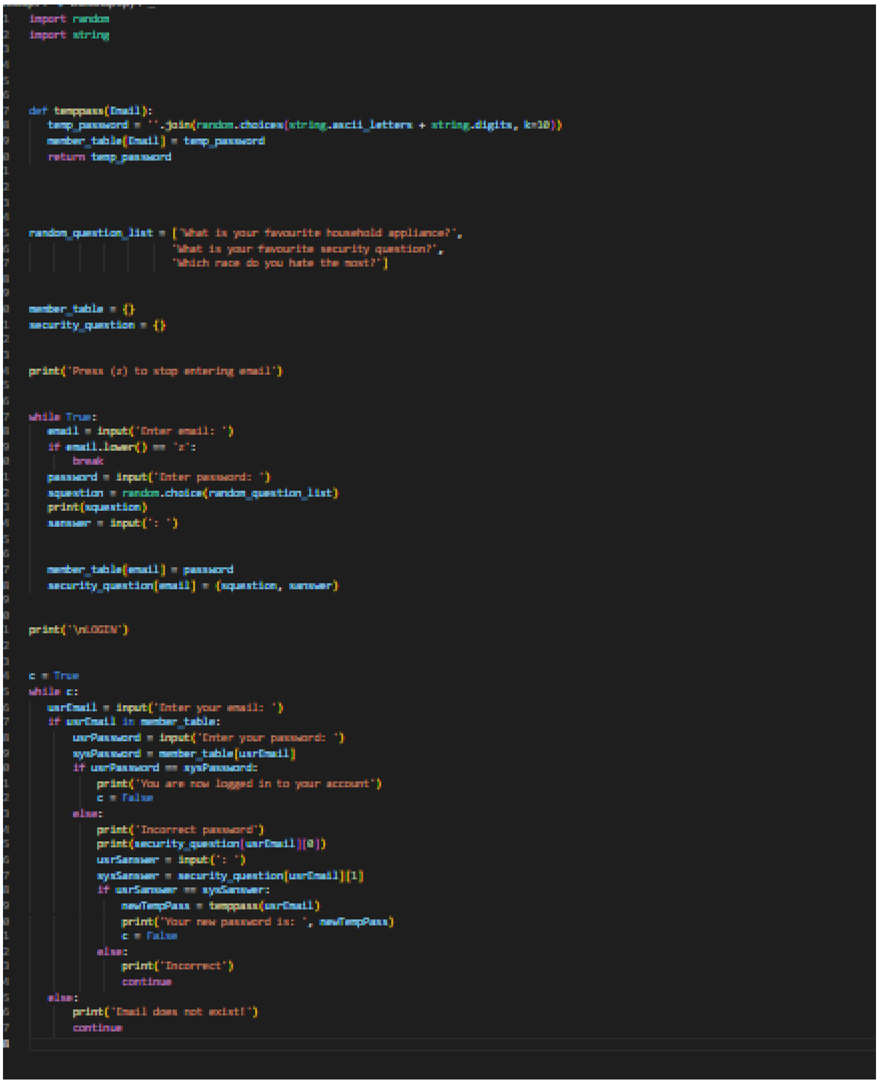
```python
assword_lookup = {}
rint('Create multiple usernames and assign passwords to each.\n'
    'Input (z) exit.')


hile True:
  usrN = input('Enter Username: ')
  if usrN.lower() == 'z':
      break
  usrP = input('Enter Password: ')
  password_lookup[usrN] = usrP


rint('You have created %d accounts.\n'
    'The accounts are %s' % (len(password_lookup), ', '.join(password_lookup.keys())))
```

5. Create a dictionary named `password_hint` that contains email addresses as keys, and associated values that contain both the users' "password security question," and the answer to the question. Make up data for dictionary entries.



```
password_hint = {}


print('Create security questions for your email.\n'
      'Enter (z) as an email to exit.')


while True:
    email = input('Enter Email: ')
    if email.lower() == 'z':
        break
    question = input('Enter Security Question: ')
    answer = input('Enter the answer to the question: ')
    password_hint[email] = (question, answer)


print('You have created %d security questions.\n'
      % (len(password_hint)))


while True:
    usrEmail = input('Enter your email: ')
    if usrEmail in password_hint:
        print(password_hint[usrEmail][0])
        usrAns = input(': ')
        if usrAns.lower() == password_hint[usrEmail][1].lower():
            print('Your account is now unlocked')
            break
        else:
            print('Incorrect answer')
            continue
    else:
        print('Email does not exist')
        continue
```

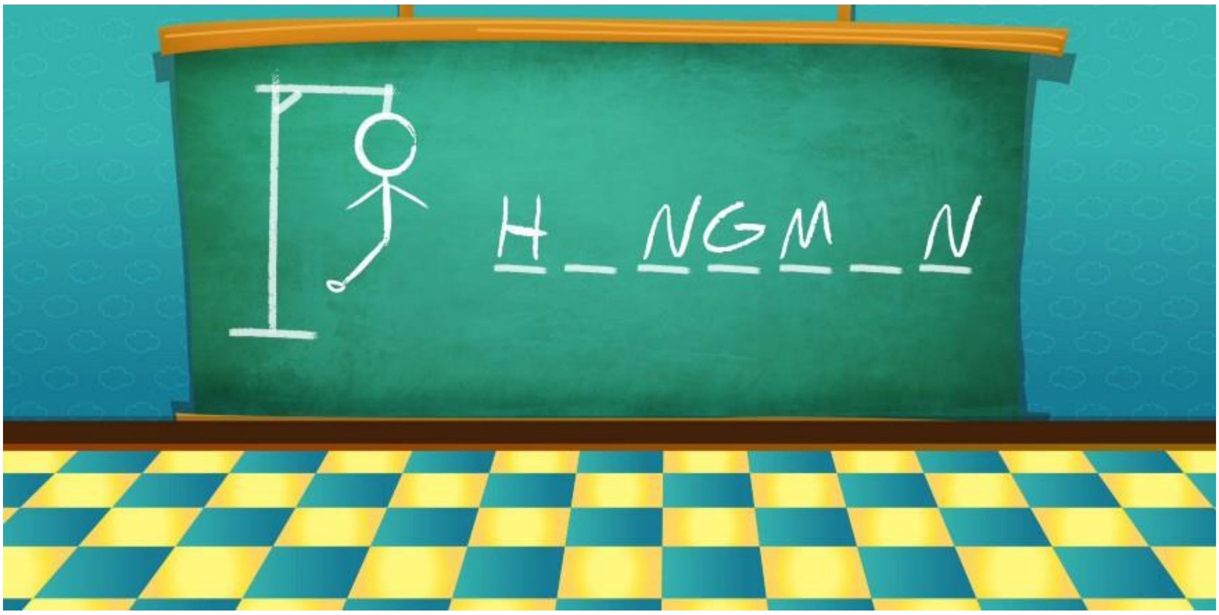
6. Create a dictionary named `member_table` that contains users' email addresses as keys, and answers to their password hints as the associated values, and a function that generates a temporary new password and stored in the table.



```python
import random
import string



def temppass(Email):
    temp_password = ''.join(random.choices(string.ascii_letters + string.digits, k=10))
    member_table[Email] = temp_password
    return temp_password




random_question_list = ['What is your favourite household appliance?',
                        'What is your favourite security question?',
                        'Which race do you hate the most?']


member_table = {}
security_question = {}


print('Press (z) to stop entering email')


while True:
    email = input('Enter email: ')
    if email.lower() == 'z':
        break
    password = input('Enter password: ')
    squestion = random.choice(random_question_list)
    print(squestion)
    sanswer = input(': ')


    member_table[email] = password
    security_question[email] = (squestion, sanswer)


print('\nLOGIN')


c = True
while c:
    usrEmail = input('Enter your email: ')
    if usrEmail in member_table:
        usrPassword = input('Enter your password: ')
        sysPassword = member_table[usrEmail]
        if usrPassword == sysPassword:
            print('You are now logged in to your account')
            c = False
        else:
            print('Incorrect password')
            print(security_question[usrEmail][0])
            usrSanswer = input(': ')
            sysSanswer = security_question[usrEmail][1]
            if usrSanswer == sysSanswer:
                newTempPass = temppass(usrEmail)
                print('Your new password is: ', newTempPass)
                c = False
            else:
                print('Incorrect')
                continue
    else:
        print('Email does not exist!')
        continue
```

Part 4 (Home Task)

1. The hangman game introduces many new concepts like *methods*, which are functions attached to values. You'll also need to learn about a data type called a *list*. Once you understand these concepts, it will be much easier to program Hangman.



1. You will need *random* module.
2. You will need to use the concept of *list.*