MTA Final Document-3

Microsoft Python Exam Preparation
Mr. Yogesh P Murumkar
Bharat Soft Solutions
Machine Learning Data Science and Big Data

Youtube – https://www.youtube.com/c/yogeshmurumkar

1

The XYZ organics company needs a simple program that their call center will use to enter survey data for a new coffee variety. The program must accept input and return the average rating based on a five-star scale.The output must be rounded to two decimal places.

Consider the code:

```
sum=count=done=0
average=0.0
while(done != -1):
        rating=float(input('Enter Next Rating(1-5),-1 for done'))
        if rating == -1:
                break
        sum+=rating
        count+=1
average=float(sum/count)
#Line-1
```

Which of the following print() statement should be placed at Line-1 to meet requirement?

- (●) `print('The average star rating for the new coffee is: {:.2f}'.format(average))` (Correct)
- ( ) `print('The average star rating for the new coffee is: {:.2d}'.format(average))`
- ( ) `print('The average star rating for the new coffee is: {:2f}'.format(average))`
- ( ) `print('The average star rating for the new coffee is: {:2.2d}'.format(average))`

Explanation

As the output required to rounded to 2 decimal places and it is float value we should use print('The average star rating for the new coffee is:{:.2f}'.format(average))

Consider the following statements:

```
1. print('V:{:.2f}'.format(123.45678)) will print to the console V:123.46
2. print('V:{:.2f}'.format(123.4)) will print to the console V:123.40
3. print('V:{:8.2f}'.format(1.45678)) will print to the console V:    1.46
4. print('V:{:08.2f}'.format(1.45678)) will print to the console V:00001.46
```

Which of the above statements are True?

- only 1 and 2
- only 1 and 3 (Incorrect)
- only 2 and 4
- 1,2,3 and 4 (Correct)

Explanation

{:.2f}==>After decimal point, the value will be rounded to 2 digits
{:8.2f}==>Total 8 length and after decimal point value will be rounded to 2 digits. Numbers are right aligned by default and vacant spaces are filled with blank space.
{:08.2f}===>Total 8 length and after decimal point value will be rounded to 2 digits. Numbers are right aligned by default and vacant spaces are filled with digit 0.

3

We are developing a sports application. Our program should allow players to enter their name and score. The program will print player name and his average score. Output must meet the following requirements:

The user name must be left aligned. If the user name is fewer than 20 characters ,additional space must be added to the right. The average score must be 3 places to the left of decimal point and one place to the right of decimal point (like YYY.Y).

Consider the code:

```
name=input('Enter Your Name:')
score=0
count=0
sum=0
while(score != -1):
        score=int(input('Enter your scores: (-1 to end)'))
        if score==-1:
                break
        sum+=score
        count+=1
average_score=sum/count
#Line-1
```

Which print statement we have to take at Line-1 to meet requirements.

- `print('%-20s,Your average score is: %4.1f' %(name,average_score))` **(Correct)**
- `print('%-20f,Your average score is: %4.1f' %(name,average_score))` **(Incorrect)**
- `print('%-20s,Your average score is: %1.4f' %(name,average_score))`
- `print('%-20s,Your average score is: %4.1s' %(name,average_score))`

Explanation

'%4.1f': Minimum 4 length
After decimal point all digits rounded to 1 digit
If the number less than 4 length then spaces will be added at left hand side '%04.1f'
Minimum 4 length
After decimal point all digits rounded to 1 digit
If the number less than 4 length then 0s will be added at left hand side
name=input('Enter Some Name:')
print('%-20s' %name)
# It will consider minimum 20 length,if it is less than 20 then spaces will be padded at right hand side
print('%20s' %name)
# It will consider minimum 20 length,if it is less than 20 then spaces will be padded at left hand side

4

Consider the following code:

```
numbers=[0,1,2,3,4,5,6,7,8,9]
index=0
while (index<10)#Line-1
   print(numbers[index])

   if numbers(index) = 6#Line-2
      break
   else:
          index += 1
```

To print 0 to 6,which changes we have to perform in the above code?

- [ ]
```
Line-1 should be replaced with
 while(index<10):
```
(Correct)

- [x]
```
Line-2 should be replaced with
  if numbers[index]==6:
```
(Correct)

- [ ]
```
Line-2 should be replaced with
 if numbers[index]=6:
```

- [ ]
```
Line-1 should be replaced with
  while(index>0):
```

Explanation

At Line-1, invalid syntax,because colon(:) is missing.
At Line-2 we should use == operator for comparison and colon(:) is missing.

5

You are writing a Python program to validate employee numbers.

The employee number must have the format dd-ddd-dddd and consists of only numbers and dashes.The program must print True if the format is correct,otherwise print False.

```
employee_number=input('Enter Your Employee Number(dd-ddd-dddd):')
parts=employee_number.split('-')
valid=False
if len(parts) == 3:
        if len(parts[0])==2 and len(parts[1])==3 and len(parts[2])==4:
                if parts[0].isdigit() and parts[1].isdigit() and parts[2].isdigit():
                        valid=True
print(valid)
```

Which of the following is True about this code

- It will throw error because misuse of split() method
- It will throw error because misuse of isDigit() method (Incorrect)
- There is no error but it won't fulfil our requirement.
- No changes are required for this code and it can fulfill requirement. (Correct)

Explanation

Every method call invoked properly and there is no error. This code can fulfill requirement without any changes.

6

Question 6: **Incorrect**

You are coding a math utility by using python.

```
You are writing a function to compute roots
The function must meet the following requirements
  If a is non-negative, return a**(1/b)
  If a is negative and even, return "Result is an imaginary number"
  if a is negative and odd,return -(-a)**(1/b)
```

Which of the following root function should be used?

```
def root(a,b):
        if a>=0:
                answer=a**(1/b)
        elif a%2 == 0:
                answer="Result is an imaginary number"
        else:
                answer=-(-a)**(1/b)
        return answer
```

(Correct)

```
def root(a,b):
        if a>=0:
                answer=a**(1/b)
        elif a%2 != 0:
                answer="Result is an imaginary number"
        else:
                answer=-(-a)**(1/b)
        return answer
```

```
def root(a,b):
        if a>=0:
                answer=a**(1/b)
        if a%2 == 0:
                answer="Result is an imaginary number"
        else:
                answer=-(-a)**(1/b)
        return answer
```

(Incorrect)

```
def root(a,b):
        if a>=0:
                answer=a**(1/b)
        elif a%2 == 0:
                answer=-(-a)**(1/b)
        else:
                answer="Result is an imaginary number"
        return answer
```

Explanation

Make sure you should remember if condition fails then only elif will be executed.

7

We are developing a online shopping application. Consider the code

```
d =input('Enter day of the week:')
discount_percentage = 3

if d== 'Monday':
        discount_percentage+=5

elif d== 'Tuesday':
        discount_percentage+=7

elif d== 'Saturday':
        discount_percentage+=10

elif d== 'Sunday':
        discount_percentage+=20
else:
        discount_percentage+=2
```

To get 5 as discount_percentage,which of the following input should be provided end user?

- Monday
- Tuesday (Incorrect)
- Thursday (Correct)
- Saturday
- Sunday

Explanation

if user enters Monday then discount_percentage will be 8
if user enters Tuesday then discount_percentage will be 10
if user enters Thursday then discount_percentage will be 5
if user enters Saturday then discount_percentage will be 13
if user enters Sunday then discount_percentage will be 23

8

We are developing gold loan application for XYZ company.

```
amount=float(input('Enter Loan Amount:'))
interest_rate=0
if amount > 0 and amount<= 50000:
        interest_rate = 10

elif amount > 50000 and amount<100000:
        interest_rate = 12

elif amount >= 100000 and amount<150000:
        interest_rate = 16

else:
        interest_rate = 22
```

For which of the following user input interest_rate will be 12.

- 50000
- 50001 (Correct)
- 100000
- 100001
- 150000

Explanation

For 50000 amount interest_rate is 10
For 50001 amount interest_rate is 12
For 100000 amount interest_rate is 16
For 100001 amount interest_rate is 16
For 150000 amount interest_rate is 22

9

We are developing an application for leave approval in XYZ Company.

```
days=int(input('Enter number of days for leave:'))
cause=input('Enter the cause:')
if days==1:
        print('Leave will be approved immediately')

elif days>1 and days<=3:
        if cause=='Sick':
                print('Leave will be approved immediately')
        else:
                print('Needs Lead Approval')

elif days>3 and days<5:
        if cause=='Sick':
                print('Needs Manager Approval')
        else:
                print('Needs Director Approval')

elif days>=5 and days<=10:
        print('Needs Director Approval')
```

In which of the following cases 'Needs Director Approval' will be printed to the console?

- `days = 2 and cause='Sick'`
- `days = 3 and cause='Personal'` **(Incorrect)**
- `days = 4 and cause='Sick'`
- `days = 4 and cause='Official'` **(Correct)**

Explanation

If days = 2 and cause='Sick' then 'Leave will be approved immediately' will be printed to the console.
If days = 3 and cause='Personal' then 'Needs Lead Approval' will be printed to the console.
If days = 4 and cause='Sick' then 'Needs Manager Approval' will be printed to the console.
If days = 4 and cause='Official' then 'Needs Director Approval' will be printed to the console.

Consider the following code:

```
marks=[30,40,50,45,50,100]
average=sum(marks)//len(marks)
grades={1:'A',2:'B',3:'C',4:'D'}
if average>=90 and average<=100:
        key=1
elif average>=80 and average<90:
        key=2
elif average>=50 and average<80:
        key=3
else:
        key=4
print(grades[key])
```

Which grade will be printed to the console?

- A
- B (Incorrect)
- C (Correct)
- D

Explanation

average=sum(marks)//len(marks)=315//6=52
For 52 average key is 3 and corresponding grade is C

11

Consider the code:

```
a=12
b=4
s='He shall not be happy if he does not work'
```

In which of the following cases result value will become 9

- `result=3 if None else a/b`
- `result=s.find('not') if s else None` **(Correct)**
- `result=s.rfind('not') if s else None`
- `result=5 if len(s)>4 else 6`

Explanation

result=3 if None else a/b==>3.0 (None is treated as False)
result=s.find('not') if s else None==>9(find() method returns the index of first match from left hand side)
result=s.rfind('not') if s else None===>33(rfind() method returns the index of first match from right hand side)
result=5 if len(s)>4 else 6===>5

We are developing loan collection agent application. Consider the code:

```
collected_amount=3000
commission=0
if collected_amount <= 2000:
        commission=50
elif collected_amount> 2500 and collected_amount<3000:
        commission=100
elif collected_amount>2500:
        commission=150
        if collected_amount>=3000:
                commission+=200
```

What will be the value of commission?

- 350 (Correct)
- 200 (Incorrect)
- 150
- 100

Explanation

As the value of collected amount is 3000 the following statements will be executed.
commission=0
commission=150
commission+=200
Hence the value of commission will become 350.

13

You are developing online shopping application.

```
Consider the code:
order_value=1500
state='ap'
delivery_charge=0
if state in ['up','mp','ts']:
        if order_value<=1000:
                delivery_charge=50
        elif order_value>1000 and order_value<2000:
                delivery_charge=100
        else:
                delivery_charge=150
else:
        delivery_charge=25
if state in ['lp','kp','ap']:
        if order_value>1000:
                delivery_charge+=20
        if  order_value<2000 and state in ['kp','ap']:
                delivery_charge+=30
        else:
                delivery_charge+=15
print(delivery_charge)
```

What is the result?

- 65 (Incorrect)
- 75 (Correct)
- 85
- 55

Explanation

As the state is 'ap' and order_value is 1500 the following lines of the code will be executed.
delivery_charge=0
delivery_charge=25
delivery_charge+=20
delivery_charge+=30
Hence the value of delivery_charge is 75.

Consider the code:

```
l=[10,20,[30,40],[50,60]]
count=0
for i in range(len(l)):
        if type(l[i])==list:
                count=count+1
print(count)
```

What is the result?

- 1
- 2 (Correct)
- 3
- 4

Explanation

list,set,tuple,dict are python's inbuilt variables to represent data structure types.

15

Consider the code:

```
l=[10,(20,),{30},{},{},[40,50]]
count=0
for i in range(len(l)):
        if type(l[i])==list:
                count+=1
        elif type(l[i])==tuple:
                count+=2
        elif type(l[i])==set:
                count+=3
        elif type(l[i])==dict:
                count+=4
        else:
                count+=5
print(count)
```

What is the result?

- 17 (Incorrect)
- 18
- 19 (Correct)
- 20

Explanation

list,set,tuple,dict are python's inbuilt variables to represent data structure types.

16

Consider the code:

```
t = (2,4,6,8,10,12)
d = {1:'A',2:'B',3:'C',4:'D',5:'E',6:'F'}
result=1
for t1 in t:
        if t1 in d:
                result+=t1
print(result)
```

What is the result?

- 12
- 13 (Correct)
- 19
- 6

Explanation

The elements 2,4,6 from the tuple present in dict as keys. Hence these elements will be added to result. Hence result is 13(=1+2+4+6)

17

Consider the code:

```
t = (2,4,6,8,10,12)
d = {1:'A',2:'B',3:'C',4:'D',5:'E',6:'F'}
result=1
for t1 in t:
        if t1 in d:
                continue
        else:
                result+=t1
print(result)
```

What is the result?

- 29
- 30 (Incorrect)
- 31 (Correct)
- 32

Explanation

The elements 8,10,12 from the tuple are not present in dict as keys. Hence these elements will be added to result. Hence result is 31(=1+8+10+12)

18

Consider the code:

```
values = [[3, 4, 5, 1], [33, 6, 1, 2]]
v = values[0][0]
for lst in values:
    for element in lst:
        if v > element:
            v = element
print(v)
```

What is the result?

- 3
- 2 (Incorrect)
- 1 (Correct)
- 4

Explanation

The above code is for finding minimum element present in the nested list.

19

Consider the code

```
def get_names():
        names=['Sunny','Bunny','Chinny','Vinny','Pinny']
        return names[2:]

def update_names(elements):
        new_names=[]
        for name in elements:
                new_names.append(name[:3].upper())
        return new_names

print(update_names(get_names()))
```

What is the result?

- ['CHI', 'VIN', 'PIN'] (Correct)
- ['VIN', 'PIN'] (Incorrect)
- ['CH', 'VI', 'PI']
- ['SU', 'BU']

Explanation

names[2:] returns all names from 2 index to end of list. i.e ['Chinny','Vinny','Pinny']. name[:3].upper() selects first 3 characters from name and will convert to upper case. Hence the output is : ['CHI', 'VIN', 'PIN']

Consider the following code

```
def my_list(x):
        lst.append(a)
        return lst

my_list('chicken')
my_list('mutton')
print(my_list('fish'))

to print the following to the console
   ['chicken','mutton','fish']
```

x should be replaced with

- a,lst=[] (Correct)
- a,lst=() (Incorrect)
- a,lst={}
- a,lst=None

Explanation

The required output is of list type and append() method is applicable only for list and hence we should take
a,lst=[]

21

Consider the following code:

```
def f1(x=0,y=0):
        return x+y
```

Which of the following method calls are valid?

- [ ] f1() (Correct)
- [x] f1('10','20') (Correct)
- [ ] f1(10) (Correct)
- [ ] f1('10')

Explanation

To use + operator both arguments should be either number type or string type.
f1()==>returns 0
f1('10','20')===>returns '1020'
f1(10)===>returns 10
f1('10')===>TypeError: must be str, not int. We cannot use + operator between '10' and 0

Consider the following code:

```
def f1(x=0,y=0):
    return x*y
```

Which of the following method calls are valid?

- [ ] f1() (Correct)
- [x] f1('10','20') (Incorrect)
- [ ] f1(10) (Correct)
- [ ] f1('10') (Correct)

Explanation

To use * operator both arguments should be either number type, or one is string type and other is int type.
Hence f1('10','20') is invalid. TypeError: can't multiply sequence by non-int of type 'str'

Consider the following code:

```
numbers=[100,20,10,70,50,60,40,30,90,80]
#Insert Code Here
print('The highest Number:{} and Least Number:{}'.format(high,low))
```

Which of the following code should be inserted to print Highest Number as 100 and Least Number as 10

```
def find_numbers():
        numbers.sort()
        return numbers[0],numbers[-1]
low,high=find_numbers()
```

(Correct)

```
def find_numbers():
        numbers.sort()
        return numbers[0],numbers[len(numbers)]
low,high=find_numbers()
```

(Incorrect)

```
def find_numbers():
        numbers.sort()
        return numbers[0],numbers[-1]
low=find_numbers()
high=find_numbers()
```

```
def find_numbers():
        numbers.sort()
        return numbers[2],numbers[0]
low,high=find_numbers()
```

Explanation

After sorting numbers list will become [10,20,30,40,50,60,70,80,90,100]. Hence numbers[0] represents low value and numbers[-1] represents highest value.

Consider the code:

```
numbers=[100,20,10,70,50,60,40,30,90,80]
def find_numbers():
        numbers.sort()
        return numbers[0],numbers[-1]

low=find_numbers()
high=find_numbers()
#Line-1
```

To print 10 100 to the console which of the following code we have to take at Line-1

- `print(low,high)`
- `print(low[0],high[-1])` (Correct)
- `print(low[-1],high[0])`
- `print(low[2],high[0])`

Explanation

In the above code, low and high are tuple type.The content of low=(10,100) and high=(10,100).Hence to print 10 100 we should use print(low[0],high[-1])

Consider the code:

```
def calculate(amount=6,factor=3):
        if amount>6:
                return amount*factor
        else:
                return amount*factor*2
```

Which of the following function calls returns 30

- `calculate()`
- `calculate(10)` (Correct)
- `calculate(5,2)`
- `calculate(1)`

Explanation

print(calculate())==>36
print(calculate(10))==>30
print(calculate(5,2))==>20
print(calculate(1))==>6

Consider the following code

```
def fib_seq(n):
        if n==0:
                return 0
        elif n==1:
                return 1
        else:
                return fib_seq(n-1)+fib_seq(n-2)
for i in range(7):
        print(fib_seq(i),end=',')
```

What is the result?

- 0,1,1,2,3,5,8, (Correct)
- 0,1,2,4,8,16,32, (Incorrect)
- 0,1,0,2,0,4,0,
- None of these

Explanation

The above program generates the first 7 fibonacci numbers. Hence the output is: 0,1,1,2,3,5,8,

You are developing a Python application for online game.

```
You need to create a function that meets the following criteria:
The function is named update_score
The function receives the current score and a value.
The function adds the value to the current score.
The function returns the new score.
```

Which of the following is valid function to fulfill this requirement?

```
update_score(score,value):
        new_score=score+value
        return new_score
```

```
def update_score(score,value):
        new_score=score+value
        return new_score
```

(Correct)

```
def update_score(score,value):
        new_score=score+value
        pass new_score
```

```
def update_score():
        new_score=score+value
        return new_score
```

Explanation

We should declare a function with def keyword. A function can return value by using return keyword. As per our requirement, compulsory the function should take some arguments and return new score. Hence the following function can fulfill our requirement. def update_score(score,value):
new_score=score+value
return new_score

The XYZ company is creating a program that allows customers to log the number of miles biked.The program will send messages based on how many miles the customer logs. Consider the following python code:

```
Line-1:
        name=input('Enter Your Name:')
        return name

Line-2:
        calories=miles*calories_per_mile
        return calories

distance=int(input('How many miles did you bike this week:'))
burn_rate=50
biker=get_name()
calories_burned=calc_calories(distance,burn_rate)
print(biker,", You burned about",calories_burned," calories")
```

The lines Line-1 and Line-2 should be replaced with:

- [ ]
```
Line-1 should be replaced with
def get_name():
```
(Correct)

- [x]
```
Line-1 should be replaced with
def get_name(name):
```
(Incorrect)

- [ ]
```
Line-1 should be replaced with
def get_name(biker):
```

- [ ]
```
Line-2 should be replaced with
def calc_calories(miles,calories_per_mile):
```
(Correct)

- [ ]
```
Line-2 should be replaced with
def calc_calories(miles,burn_rate):
```

- [ ]
```
Line-2 should be replaced with
def calc_calories():
```

Explanation

The following are valid function declarations to fulfill our requirement: def get_name():
name=input('Enter Your Name:')
return name
def calc_calories(miles,calories_per_mile):
calories=miles*calories_per_mile
return calories

29

You work for a company that distributes media for all ages.

You are writing a function that assigns a rating based on a user's age.The function must meet the following requirements.

Anyone 18 years old or older receives a rating of "A"

Anyone 13 or older,but younger than 18, receives a rating of "T"

Anyone 12 years old or younger receives a rating of "C"

If the age is unknown ,the rating is set to "C"

Which of the following code meets above requirements:

```
def get_rating(age):
        if age>=18:
                rating="A"
        elif age>=13:
                rating="T"
        else:
                rating="C"
        return rating
```

(Correct)

```
def get_rating(age):
        if age>=18:
                rating="A"
        if age>=13:
                rating="T"
        else:
                rating="C"
        return rating
```

(Incorrect)

```
def get_rating(age):
        if age>18:
                rating="A"
        elif age>13:
                rating="T"
        else:
                rating="C"
        return rating
```

```
def get_rating(age):
        if age>=18:
                rating="A"
        elif age>=13:
                rating="T"
        else:
                rating="C"
        pass rating
```

Explanation

The correct function to fulfill above requirements is
def get_rating(age):
if age>=18:
rating="A"
elif age>=13:
rating="T"
else:
rating="C"
return rating

Consider the following Python Code:

```
def count_letter(letter,word_list):
        count=0
        for word in word_list:
                if letter in word:
                        count +=1
        return count
word_list=['apple','pears','orange','mango']
letter=input('Enter some alphabet symbol:')
letter_count=count_letter(letter,word_list)
print(letter_count)
```

If the user provides input 'a' then what is the result?

- 1
- 2 (Incorrect)
- 3
- 4 (Correct)

Explanation

The above program prints the number of occurrences of specified alphabet symbol in the given word_list.

31

Consider the code:

```
startmsg = "hello"
endmsg = ""
for i in range(0,len(startmsg)):
        endmsg = startmsg[i] + endmsg
print(endmsg)
```

What is the result?

- olleh (Correct)
- hello (Incorrect)
- IndexError
- hlelo

Explanation

It is the program to reverse content of the given String.

32

Consider the code:

```
x = [13,4,17,10]
w = x[1:]
u = x[1:]
y = x
u[0] = 50
y[1] = 40
print(x)
```

What is the result?

- [13, 40, 17, 10] (Correct)
- [50, 40, 10] (Incorrect)
- [13,4,17,10]
- [50,40,17,10]

Explanation

In the above example both x and y are pointing to the same object.By using y if we are changing the content the changes will be reflected to x. By using slice operator a new object will be created.

You want to add comments to your code so that other team members can understand it.

What should you do?

- Place the comments after the #sign on any line **(Correct)**
- Place the comments after the last line of the code separated by a blank line **(Incorrect)**
- Place the comments before the first line of code separated by a blank line
- Place the comments inside parentheses anywhere

Explanation

If any line starts with #sign then it acts as python single line comment.

We are creating a function to calculate the power of a number by using python.

We have to ensure that the function is documented with comments.

Consider the code(Line numbers included for reference):

```
01 # The calc_power function calculates exponents
02 # x is the base
03 # y is the exponent
04 # The value of x raided to the y power is returned
05 def calc_power(x, y):
06       comment="#Return the value"
07       return x**y #raise x to the power y
```

Which of the following statements are true?

- (●) Lines 01 through 04 will be ignored for syntax checking **(Correct)**
- ( ) The hash sign(#) is optional for lines 01 and 03.
- ( ) The String in line 06 will be interpreted as a comment

Explanation

If any line starts with #sign then it acts as python single line comment.

You are writing a python script to convert student marks into grade. The grades are defined as follows:

```
90 through 100 --> A grade
80 through 89 --> B grade
70 through 79 --> C grade
65 through 69 --> D grade
0 through 64 --> E grade
```

And developed application is :

```
# Grade Converter
marks=int(input('Enter Student Marks:'))
if marks >=90: #Line-1
        grade='A'
elif marks>=80: #Line-2
        grade='B'
elif marks>=70: #Line-3
        grade='C'
elif marks>=65:
        grade='D'
else:
        grade='E'
print('Your grade is:',grade)
```

Which of the following changes should be performed to fulfill the requirement?

- ( ) Line-1 should be replaced with
  `if marks <= 90:`
- (•) Line-2 should be replaced with
  `if marks>=80 and marks <= 90 :` (Incorrect)
- ( ) Line-3 should be replaced with
  `if marks>=70 and marks <= 80 :`
- ( ) No Changes are required. (Correct)

Explanation

No changes are required for the above program. Make sure you should remember if condition fails then only elif will be executed.

You are developing a Python application for an online product distribution company. You need the program to iterate through a list of products and escape when a target product ID is found.

Which of the following code can fulfill our requirement

```
productIdList=[0,1,2,3,4,5,6,7,8,9]
index=0
while index<len(productIdList):
        print(productIdList[index])
        if productIdList[index]==6:
                break
        else:
                index+=1
```

(Correct)

```
productIdList=[0,1,2,3,4,5,6,7,8,9]
index=1
while index<len(productIdList):
        print(productIdList[index])
        if productIdList[index]==6:
                break
        else:
                index+=1
```

(Incorrect)

```
productIdList=[0,1,2,3,4,5,6,7,8,9]
index=0
while index<len(productIdList):
        print(productIdList[index])
        if productIdList[index]==6:
                continue
        else:
                index+=1
```

```
productIdList=[0,1,2,3,4,5,6,7,8,9]
index=0
while index<len(productIdList):
        print(productIdList[index])
        if productIdList[index]==6:
                continue
        else:
                index+=1
```

```
productIdList=[0,1,2,3,4,5,6,7,8,9]
index=0
while index<len(productIdList):
        print(productIdList[index])
        if productIdList[index]==6:
                break
        else:
                continue
```

Explanation

While developing the application, we have to consider the following syntactical things.
The body of the while loop will be executed as long as condition is True.
Inside while loop to break loop execution based on some condition, we should go for break statement. The index of the first element inside list is 0.
Hence the following code fulfills our requirement
productIdList=[0,1,2,3,4,5,6,7,8,9]
index=0
while index print(productIdList[index])
if productIdList[index]==6:
break
else:
index+=1

You are writing a python program that displays all prime numbers from 2 to 200. Which of the following is the proper code to fulfill our requirement?

```
p=2
while p<=200:
        is_prime=True
        for i in range(2,p):
                if p % i == 0:
                        is_prime=False
                        break

        if is_prime==True:
                print(p)
        p=p+1
```

(Correct)

```
p=2
is_prime=True
while p<=200:
        for i in range(2,p):
                if p % i == 0:
                        is_prime=False
                        break

        if is_prime==True:
                print(p)
        p=p+1
```

(Incorrect)

```
p=2
while p<=200:
        is_prime=True
        for i in range(2,p):
                if p % i == 0:
                        is_prime=False
                        break

        if is_prime==False:
                print(p)
        p=p+1
```

```
p=2
while p<=200:
        is_prime=True
        for i in range(2,p):
                if p % i == 0:
                        is_prime=False
                        break

        if is_prime==True:
                print(p)
```

Explanation

A positive integer greater than 1 which has no other factors except 1 and the number itself is called a prime number.
2, 3, 5, 7 etc. are prime numbers as they do not have any other factors. But 6 is not prime (it is composite) since, 2 x 3 = 6.
The following code fulfills our requirement
p=2
while p<=200:
is_prime=True
for i in range(2,p):
if p % i == 0:
is_prime=False
break
if is_prime==True:
print(p)
p=p+1

You created the following program to locate a conference room and display room name.

```
1  rooms={1:'Left Conference Room',2:'Right Conference Room'}
2  room=input('Enter the room number:')
3  if  not room in rooms:#Line-3
4          print('Room does not exist')
5  else:
6          print('The room name is:'+rooms[room])
```

team reported that the program sometimes produces incorrect results.

You need to troubleshoot the program. Why does Line-3 Fails to find the rooms?

- ○ Invalid Syntax **(Correct)**
- ◉ Mismatched data type(s) **(Incorrect)**
- ○ Misnamed variable(s)
- ○ None of these

Explanation

To meet the requirement we have to write Line-3 as if room not in rooms:

The XYZ Book Company needs a way to determine the cost that a student will pay for renting a Book.

The Cost is dependent on the time of the Book is returned.

However there are also special rates on Saturday and Sundays.

The Fee Structure is shown in the following list:

The cost is $3.00 per night.

If the Book is returned after 9PM, the student will be charged an extra day.

If the Book is rented on a Sunday,the student will get 50% off for as long as they the keep the book.

If the Book is rented on a Saturday,the student will get 30% off for as long as they keep the book.

We need to write the code to meet this requirements.

```
# XYZ Book Rented Amount Calculator
ontime=input('Was Book returned before 9 pm? y or n:').lower()
days_rented=int(input('How many days was book rented?'))
day_rented=input('What day the Book rented?').capitalize()
cost_per_day=3.00
if ontime == 'n':
        days_rented=days_rented+1
if day_rented=='Sunday':
        total=(days_rented*cost_per_day)*0.5
elif day_rented=='Saturday':
        total=(days_rented*cost_per_day)*0.7
else:
        total=(days_rented*cost_per_day)
print('The Cost of Book Rental is:$',total)
```

If the Book rented on 'Sunday',the number of days Book rented is 5 and Book returned after 9PM then what is the result?

- The Cost of Book Rental is:$ 7.0

- The Cost of Book Rental is:$ 7.0
- The Cost of Book Rental is:$ 8.0 (Incorrect)
- The Cost of Book Rental is:$ 9.0 (Correct)
- The Cost of Book Rental is:$ 10.0

Explanation

If the Book rented on 'Sunday',the number of days Book rented is 5 and Book returned after 9PM then the following lines will be executed.
cost_per_day=3.00
days_rented=days_rented+1
total=(days_rented*cost_per_day)*0.5
Hence the output will become 9.0 total.

40

We are developing one school automation application. If student marks between 80 and 100,then we have to offer 'A' grade.

Which code block we have to use?

```
if marks>=80 and marks>=100:
        grade='A'
```

```
if marks>=80 or marks<=100:
        grade='A'
```

```
if 80<=marks<=100:
        grade='A'
```
(Correct)

```
if marks>80:
        grade='A'
```

Explanation

Nesting of relational operators is allowed.
a<=b<=c
In Nesting, if all conditions are True then only result is True. If atleast one condition fails then the result is False
if 80<=marks<=100:
grade='A'
If marks >=80 and marks<=100 then only grade will become 'A', which will meet our requirement.