***About Python***

*In order to tell the computer „what you want to do‟, we write a program in a language which computer can understand. Though there are many different programming languages such as BASIC, Pascal, C, C++, Java, Haskell, Ruby, Python, etc.*

*Python was created by Guido Van Rossum when he was working at CWI (Centrum Wiskunde & Informatica) which is a National Research Institute for Mathematics and Computer Science in Netherlands. The language was released in I991. Python got its name from a BBC comedy series from seventies- “Monty Python‟s Flying Circus”.*

*Python can be used to follow both Procedural approach and Object Oriented approach*

*of programming. It is free to use.*

***Some of the features which make Python so popular are as follows:***

* It is a general purpose programming language which can be used for both scientific and non scientific programming.*

* It is a platform independent programming language.*

* It is a very simple high level language with vast library of add-on modules.*

* It is excellent for beginners as the language is interpreted, hence gives immediate results.*

* The programs written in Python are easily readable and understandable.*

* It is suitable as an extension language for customizable applications.*

* It is easy to learn and use.*

***The language is used by companies in real revenue generating products, such as:***

* In operations of Google search engine, youtube, etc.*

* Bit Torrent peer to peer file sharing is written using Python*

* Intel, Cisco, HP, IBM, etc use Python for hardware testing.*

* Maya provides a Python scripting API*

* i–Robot uses Python to develop commercial Robot.*

* NASA and others use Python for their scientific programming task.*

***Data types in Python***

*1.* ***Number:*** *Number data type stores Numerical Values. This data type is immutable i.e. value of its object cannot be changed. Numbers are of three different types:*

*Integer & Long (to store whole numbers i.e. decimal digits without fraction part)*

*Float/floating point (to store numbers with fraction part)*

*Complex (to store real and imaginary part)*

*2.* ***None:*** *This is special data type with a single value. It is used to signify the absence of value/false in a situation. It is represented by None.*

*3.* ***Sequence:*** *A sequence is an ordered collection of items, indexed by positive integers. It is a combination of mutable (a mutable variable is one, whose value may change) and immutable (an immutable variable is one, whose value may not change) data types. There are three types of sequence data type available in Python, they are Strings, Lists & Tuples.*

*3.1* ***String-*** *is an ordered sequence of letters/characters. They are enclosed in single quotes (' ') or double quotes ('' "). The quotes are not part of string. They only tell the computer about where the string constant begins and ends. They can have any character or sign, including space in them. These are* ***immutable.*** *A string with length 1 represents a character in Python.*

*3.2* ***Lists:*** *List is also a sequence of values of any type. Values in the list are called elements / items. These are* ***mutable*** *and indexed/ordered. List is enclosed in square brackets ([]).*

*3.3* ***Tuples:*** *Tuples are a sequence of values of any type and are indexed by integers. They are immutable. Tuples are enclosed in ().*

*4.* ***Sets:*** *Set is unordered collection of values of any type with no duplicate entry. It is immutable.*

*5.* ***Mapping:*** *This data type is unordered and mutable. Dictionaries fall under Mappings.*

*5.1* ***Dictionaries:*** *It can store any number of python objects. What they store is a* ***key -value*** *pairs,*

*which are accessed using key. Dictionary is enclosed in curly brackets ({}).*

*C.* ***Value:*** *Value is any number or a letter or string. To bind value to a variable, we use assignment*

*operator (=).*

***Data Types***

*Numbers None Sequences Sets Mappings*

*Dictionary*

*Integer Strings Tuple List*

*Boolean*

*Floating Complex*

*Point*

*Computer Science*

***Keywords*** *- are used to give some special meaning to the interpreter and are used by Python interpreter to recognize the structure of program. A partial list of keywords in Python 2.7 is*

*and del from not*

*while as elif global*

*or with assert else*

*if pass Yield break*

*except import print class*

*exec in Raise continue*

*finally is return def*

*for lambda try*

***OBJECT OREIENTED PROGRAMMING***

*An object-oriented programming (OOP) is a programming language model which is organized around "objects" rather than "actions" and data rather than logic. Before the introduction of the Object Oriented Programming paradigm, a program was viewed as a logical procedure that takes input data, processes it, and produces output. But in case of OOP a problem is viewed in terms of objects rather than procedure for doing it. Now the question arises what is an object?*

*An object can be anything that we notice around us. It can be a person (described by name, address, date of Birth etc, his typing speed), a cup (described by size , color , price etc.) , a car (described by model , color , engine etc., its mileage, speed ) and so on. In fact it can be an identifiable entity. The whole idea behind an object oriented model is to make programming closer to they real world thereby making it a very natural way of programming. The core of pure object-oriented programming is to combine into a single unit both data and functions or methods that operate on that data.*

*Simula was the first object-oriented programming language. Java, Python, C++, Visual Basic, .NET and*

*Ruby are the most popular OOP languages today.*

***The basic concepts related to OOP are as follows:***

*1. Objects*

*2. ClassesComputer S*

*3. Encapsulation*

*4. Abstraction*

*5. Data Hiding*

*6. Polymorphism*

*7. Inheritance*

***Object:***

*An object is the basic key concept of Object Oriented Programming. As mentioned before it can be anything around us - a person, place, any activity or any other identifiable entity.*

***Classes***

*A class is group of objects with same attributes and common behaviours. It is basically a blueprint to create objects. An object is a basic key concept of OOP but classes provide an ability to generalize similar type of objects. Both data and functions operating on the data are bundled as a unit in a class for the same category of objects.*

***Encapsulation***

*Encapsulation is the most basic concept of OOP. It is the combining of data and the functions associated with that data in a single unit. In most of the languages including python, this unit is called a class.*

*In simple terms we can say that encapsulation is implemented through classes. In fact the data members of a class can be accessed through its member functions only. It keeps the data safe from any external interference and misuse. The only way to access the data is through the functions of the class.*

***Data Hiding***

*Data hiding can be defined as the mechanism of hiding the data of a class from the outside world or to be precise, from other classes. This is done to protect the data from any accidental or intentional access.*

*In most of the object oriented programming languages, encapsulation is implemented through classes. In a class, data may be made private or public. Private data or function of a class cannot be accessed from outside the class while public data or functions can be accessed from anywhere. So data hiding is achieved by making the members of the class private. Access to private members is restricted and is only available to the member functions of the same class. However the public part of the object is accessible outside the class.*

***Abstraction***

*The process of identifying and separating the essential features without including the internal details is abstraction. Only the essential information is provided to the outside world while the background details are hidden. Classes use the concept of abstraction. A class encapsulates the relevant data and functions that operate on data by hiding the complex implementation details from the user. The user needs to focus on what a class does rather than how it does.*

***Inheritance***

*Inheritance is one of the most useful characteristic of object-oriented programming as it enforces reusability of code. Inheritance is the process of forming a new class (derived class) from an existing class (called the base class). The data members and the methods associated with the data are accessible in the inherited class.*

***Polymorphism***

*The word Polymorphism is formed from two words - poly and morph where poly means many and morph means forms. So polymorphism is the ability to use an operator or function in various forms. That is a single function or an operator behaves differently depending upon the data provided to them.*

***Static and Dynamic Binding***

*Binding is the process of linking the function call to the function definition. The body of the function is executed when the function call is made. Binding can be of two types:*

***Static Binding:*** *In this type of binding, the linking of function call to the function definition is done during compilation of the program.*

***Dynamic Binding:*** *In this type of binding, linking of a function call to the function definition is done at run time. That means the code of the function that is to be linked with function call is unknown until it is executed. Dynamic binding of functions makes the programs more flexible.*

***Advantages of OOP***

*Object Oriented programming has following advantages:*

***Simplicity:*** *The objects in case of OOP are close to the real world objects, so the complexity of the program is reduced making the program structure very clear and simple.*

***Modifiability:*** *It is easy to make minor changes in the data representation or the procedures in an OO program. Changes inside a class do not affect any other part of a program, since the only public interface that the external world has to a class is through the use of methods.*

***Extensibility and Maintainability:*** *It is quite easy to add new features and extend the program in case of object oriented programming. It can be simply done by introducing a few new objects and modifying some existing ones. The original base class need not be modified at all. Even objects can be maintained separately. There by making locating and fixing problems easier. For example if a new version of i-phone is introduced, a new derived class of the class i\_phone for the new version may be created and no other class in the class hierarchy need to be modified. Similarly if any behaviour of a Windows phone changes, maintenance has to be done only for the class Windows phone.*

***Re-usability:*** *Objects can be reused in different programs. The class definitions can be reused in various applications. Inheritance makes it possible to define subclasses of data objects that share some or all of the main class characteristics. It forces a more thorough data analysis, reduces development time, and ensures more accurate coding.*

***Security:*** *Since a class defines only the data it needs to be concerned with, when an instance of that class (an object) is run, the code will not be able to accidentally access other program data. This characteristic of data hiding provides greater system security and avoids unintended data corruption.*

***Data File Handling***

*A file (i.e. data file) is a named place on the disk where a sequence of related data is stored. In python files are simply stream of data, so the structure of data is not stored in the file, along with data. Basic operations performed on a data file are:*

*Naming a file*

*Opening a file*

*Reading data from the file*

*Writing data in the file*

*Closing a file*

*Using these basic operations, we can process file in many ways, such as*

*Creating a file*

*Traversing a file for displaying the data on screen*

*Appending data in file*

*Inserting data in file*

*Deleting data from file*

*Create a copy of file*

*Updating data in the file, etc.*

*Python allow us to create and manage two types of file*

*Text*

*Binary*

*A text file is usually considered as sequence of lines. Line is a sequence of characters (ASCII), stored on permanent storage media. Although default character coding in python is ASCII but using constant u with string, supports Unicode as well. As we talk of lines in text file, each line is terminated by a special character, known as End of Line (EOL). From strings we know that \n is newline character. So at the lowest level, text file will be collection of bytes. Text files are stored in human readable form and they can also be created using any text editor.*

*A binary file contains arbitrary binary data i.e. numbers stored in the file, can be used for numerical operation(s). So when we work on binary file, we have to interpret the raw bit pattern(s) read from the file into correct type of data in our program. It is perfectly possible to interpret a stream of bytes originally written as string, as numeric value. But we know that will be incorrect interpretation of data and we are not going to get desired output after the file processing activity. So in the case of binary file it is extremely important that we interpret the correct data type while reading the file. Python provides special module(s) for encoding and decoding of data for binary file.*

*To handle data files in python, we need to have a file object. Object can be created by using open() function or file() function. To work on file, first thing we do is open it. This is done by using built in function open(). Using this function a file object is created which is then used for accessing various methods and functions available for file manipulation*

***Exception Handling***

*When we plan our code/program, we always work for situations that are normally expected, and our program works very well in those situations. But, we all understand that programs have to deal with errors. Here errors are not syntax errors instead they are the unexpected condition(s) that are not part of normal operations planned during coding. Partial list of such kinds of errors are:*

*Out of Memory*

*Invalid filename*

*Attempting to write into read only file*

*Getting an incorrect input from user*

*Division by zero*

*Accessing an out of bound list element*

*Trying to read beyond end of file*

*Sending illegal arguments to a method*

*If any of such situations is encountered, a good program will either have the code check for them and perform some suitable action to remedy them, or at least stop processing in a well defined way after giving appropriate message(s). So what we are saying is if an error happened, there must be code written in program, to recover from the error. In case if it is not possible to handle the error then it must be reported in user friendly way to the user.*

*Errors are exceptional, unusual and unexpected situations and they are never part of the normal flow of a program. We need a process to identify and handle them to write a good program. Exceptions handling is the process of responding in such situations. Most of the modern programming languages provide support with handling exceptions. They offer a dedicated exception handling mechanism, which simplifies the way in which an exception situation is reported and handled.*

*Before moving ahead with exception handling, let's understand, some of the terms associated with it - when an exception occurs in the program, we say that exception was raised or thrown. Next, we deal with it and say it is handled or caught. And the code written to handle it is known as exception handler.*

*For handling exceptional situations python provides*

*1.* ***raise statement*** *to raise exception in program*

*2.* ***try..... except*** *statement for catching and handling the errors.*

*Raise statement allows the programmer to force a specified exception to occur. Once an exception is raised,*

*it's up to caller function to either handle it using try/except statement or let it propagate further.*

***Linear List Manupluation***

***Data Structures***

*A data structure is a group of data which can be processed as a single unit. This group of data may be of similar or dissimilar data types. Data Structures are very useful while programming because they allow processing of the entire group of data as a single unit. This makes managing of data simpler. The simplest form of data structure is an array which combines finite data of same data type.*

*Data structures are of two types: Linear and Non - Linear. In a linear data structure, the elements are stored in a sequential order. On the other hand, in a non linear data structure no sequential order is followed. It is a sort of multilevel data structure. Arrays, lists, stacks, queues, linked lists etc. are examples of linear data structure while tree, graph etc. is a non - linear data structure.*

*List is one of the simplest and most important data structures in Python. A list is a sequence of values of any data type. These ordered set of values are called elements or members of a list and are enclosed in square brackets[ ]. To identify a value of a list, we use index.*

***STACK AND QUEQUE IN LIST***

*STACK*

*A stack is a data structure whose elements are accessed according to the Last-In First-Out (LIFO) principle. This is because in a stack, insertion and deletion of elements can only take place at one end, called top of the stack. Consider the following examples of stacks:*

*1. Ten glass plates placed one above another. (The plate that is kept last has to be taken out first)*

*2. The tennis balls in a container. (You cannot remove more than one ball at a time)*

*3. A pile of books*

*4. A stack of coins*

*The two operations performed on the stack are:*

*1. Push: Adding(inserting) new element on to the stack.*

*2. Pop: Removing (deleting) an element from the stack*

*QUEQUE*

*Another most common data structure found in computer algorithm(s) is queue. We are already familiar with it, as we run into enough of them in our day to day life. We queue up\_\_*

*at the bank*

*at fee counter*

*at shopping centre etc.*

*A queue is a container of elements, which are inserted and removed according to the first-in first-out (FIFO) principle.*

*In a queue, persons who stand in the queue will carry out their work one by one. That means those who*

*stands first in the queue will be allowed to carry out his work first and the person who stands at the second*

*position will be allowed to carry out his work second only. At the same time those who come late will be*

*joining the queue at the end. In simple terms it is called 'first come first out'.*

*Technically speaking a queue is a linear list, to keep an ordered collection of elements / objects. The*

*principle operations, which can be performed on it are*

*Addition of elements &*

*Removal of elements.*

***Project Overview***

My project *“Online Examination”* is a just a step forward to a new concept recently prevailing in our country. This project involves many features to the user. It rotates around the needs of the user.

Now-a-days many competitive exams had become online so its programming is a major concern, in which I had worked on. One should not feel uncomfortable or distressed regarding online examination, it should be as free as pen-paper examination. We have focused a lot in that issue and tried to overcome it.

Besides all this, learning is also important. Many topics of class 12th and 11th are covered briefly. Concept are more cleared. Application of theory is done to an great extent.

**Salient Features about Project**

1) My project is divided into one main module and two sub-module:-

\* Main Module:- Online Examination.py

\* Sub Module:- 1) Adminfunction.py

2) Userfunction.py

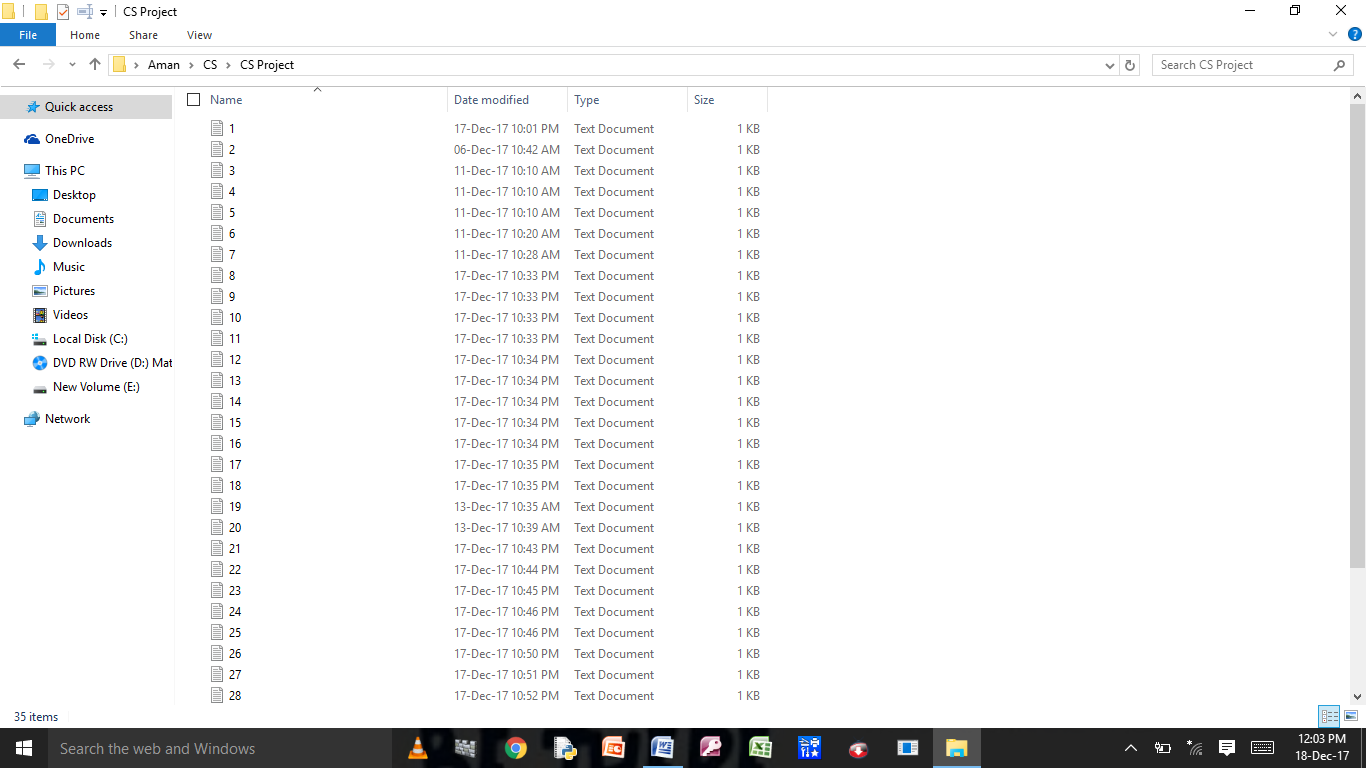
Main module in which the whole programming was done comprises of both sub module i.e. Adminfunction.py and Userfunction.py. A algorithmic programming is done in it so that while executing a user cannot feel it such complicated.

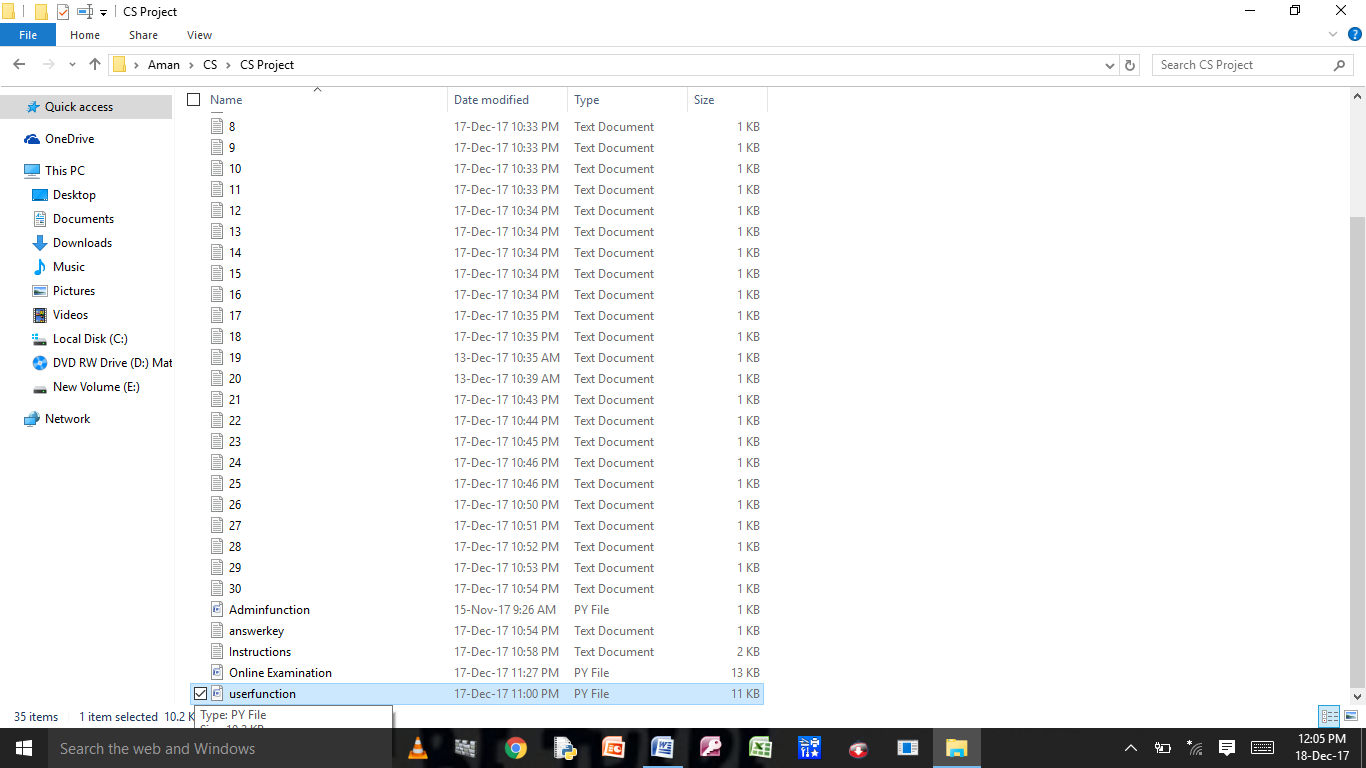
Hence for a user it is very easy to use. My project is mainly for two communities that is one who is conducting the exam and other who is giving the examination. Main Module comprises of both of them i.e both of the communities can go for it together.

On the other hand both the sub-module provide a systematic login to both of them. They can login into it separately so that there will be no issue of any type of problem. But as recommended by us we prefer Main Module for execution.

2) Since my project is about Online Examination there is a need of questions files, hence the use of file handling is the main approach of me towards my project.

Each question file is made separately for proper execution of my project (this will be discussed in next few points). My Examination covered of 30 questions and no doubt 30 questions files are made, also a combines question file is made for the admin to look out together all questions.





*3) ADMIN Module*

This mainly comprises all the feature for admin i.e. it gives complete information about the exam which is going to be conducted by him. It is not that much flexible but provide all the necessary information about the exam.

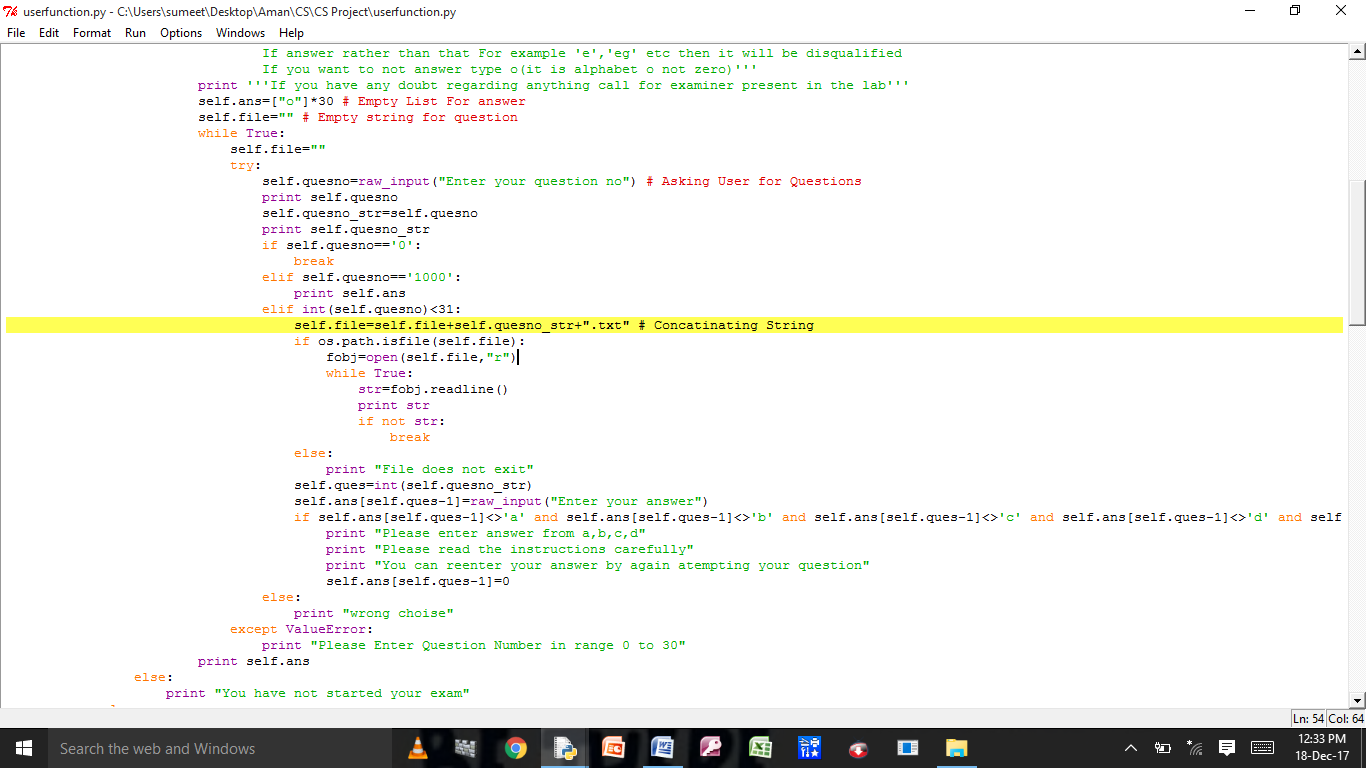
Main use of file handling is done. A feature of directly open a file to a word format is done.

*4) USER Module*

For me this is the backbone of our project. After completing the project when I look it really feels that yes it can be very versatile for the user who is giving the exam.

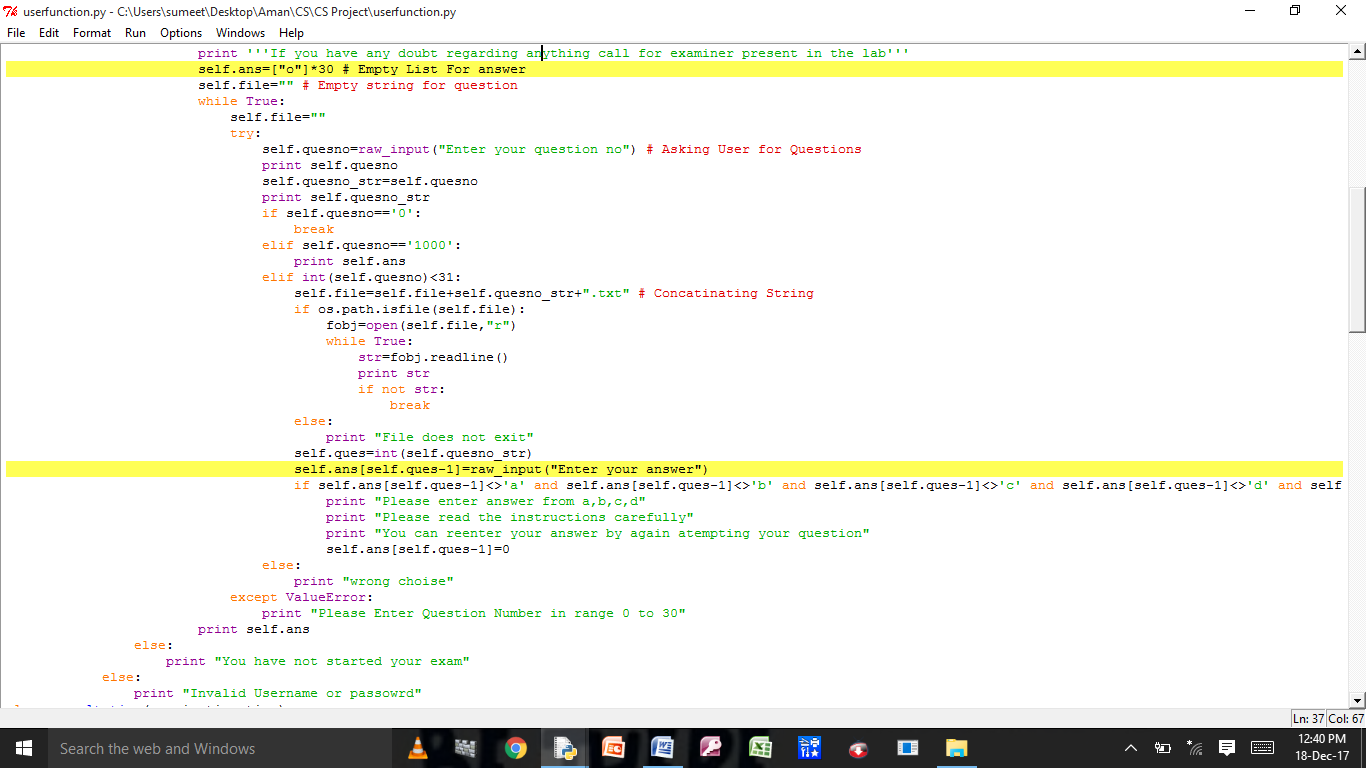
Use of file handling, object oriented programming and a crucial role of looping is seen in it. Concept of looping is the main feature of this module or I can say that main feature of my project.

* Starting with Object oriented programming , two classes named examination\_time and result\_time are made and also inherited. Inheritation of result\_time is done with the examination\_time. Both the classes comprises of one function user( ) in classs examination\_time and count\_marks( ) in result\_time.
* Coming to File Handling , I have already told that as per my topic use of it is very common. But in this module I have use it several times and also in many different ways.

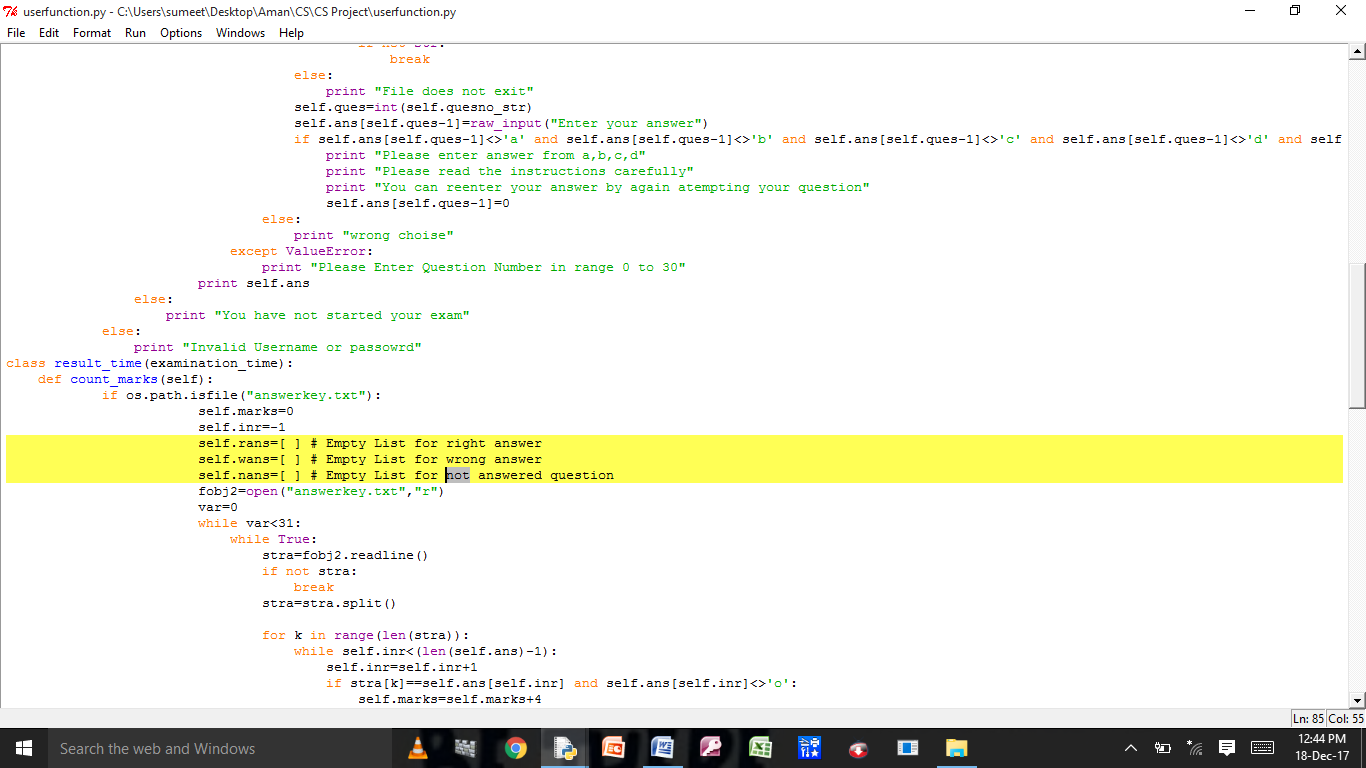


*This highlighted portion can clearly explained what I am trying to say. Concept of file handling is used with looping.*

* Next is concept of list in this module. *It is one of the hidden feature in my project* which gives a new shine to my project. It is also used in many ways.



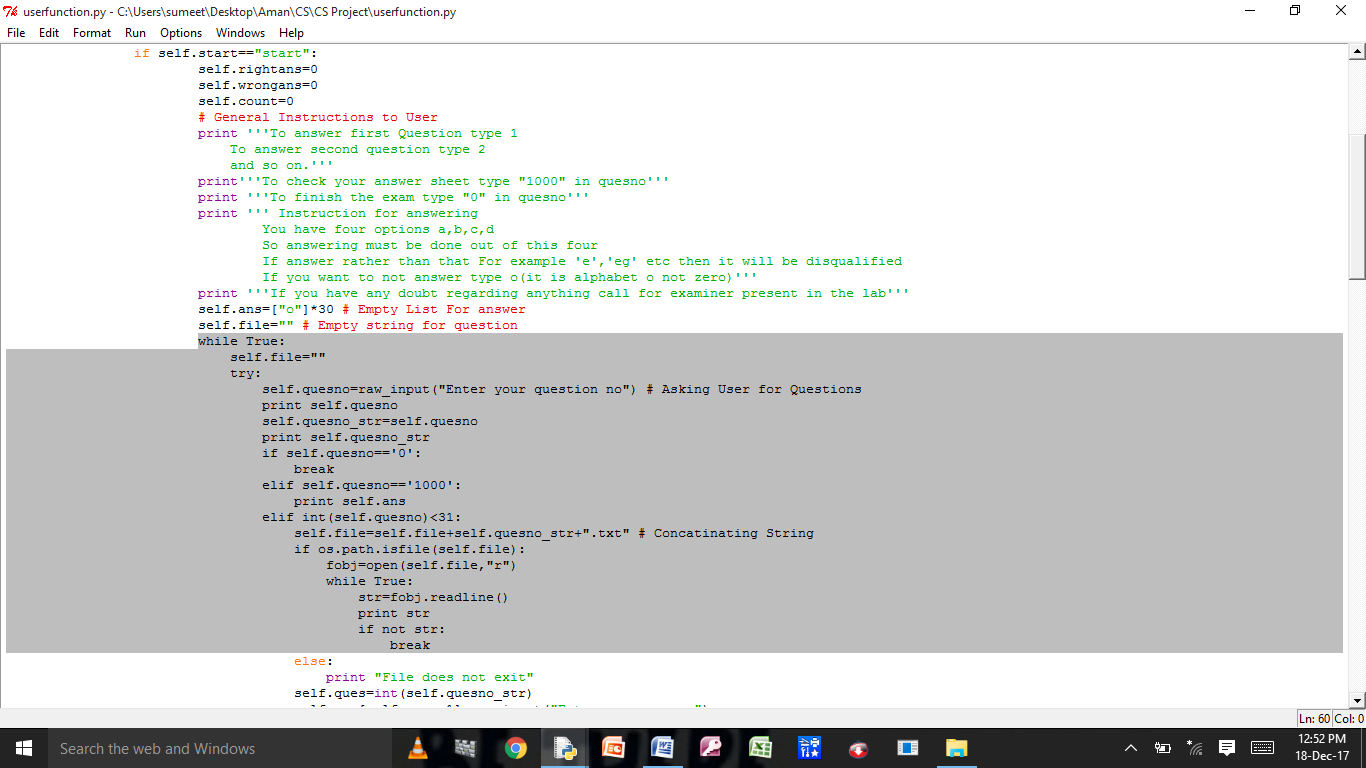
In this instead of appending the values, I have replaced it so that my program will not become complicated. It is not so that I have not tried to append the values. Once I have succeeded but I have become complicated in itself only. So this concept of list is used.



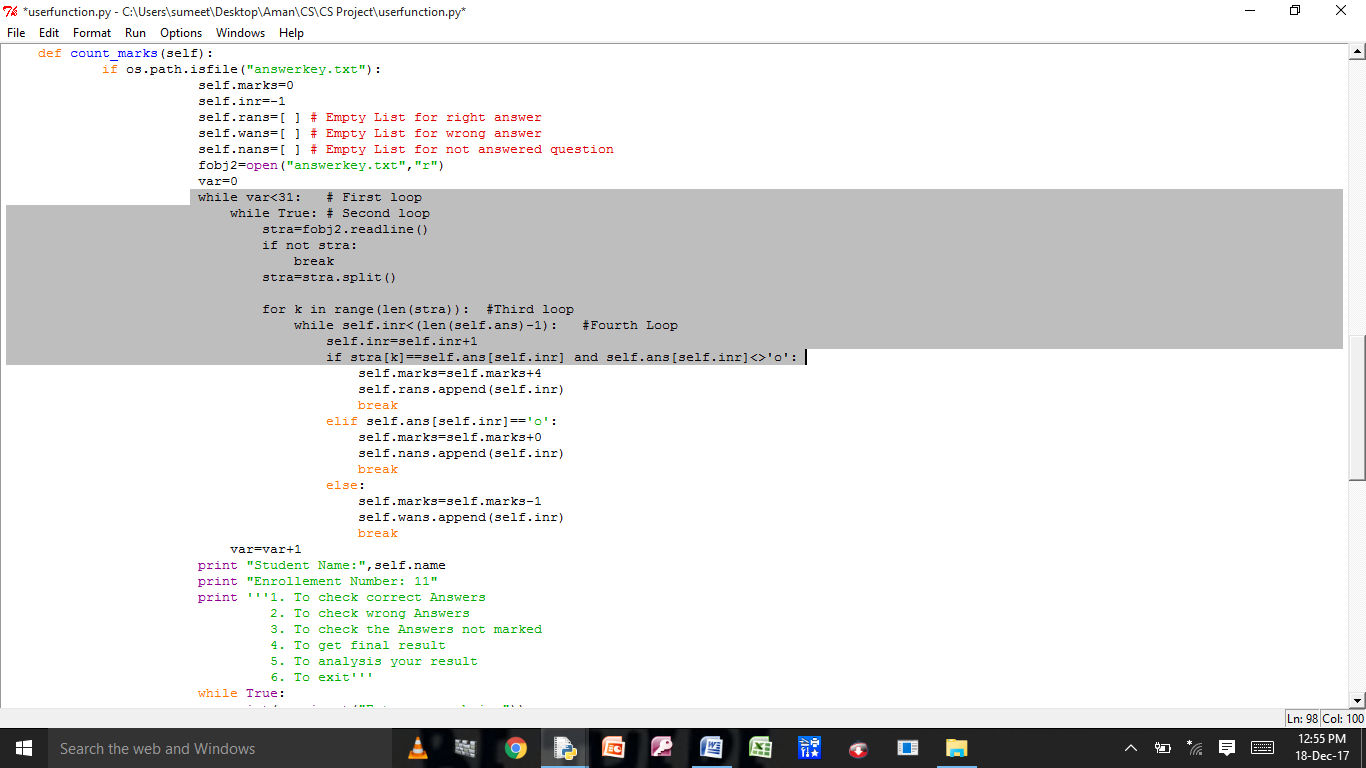
Simple appending of list is also done as shown above. Hence role of list is also very important in my project.

* Role of ‘if’: In any programming role of ‘if’ is like water in the earth. It is impossible for me to make my project without if. It is main tool for me to execute my program. Like list it is also a hidden feature of my project.
* Looping: I have learnt a lot in this project about looping. It looks so simple to use but in fact it is main challenge for me to execute it properly. My 60% of problems arises due to looping. On the other handing this problems only taught me a lot.

Looping is also used in many different ways in my project.



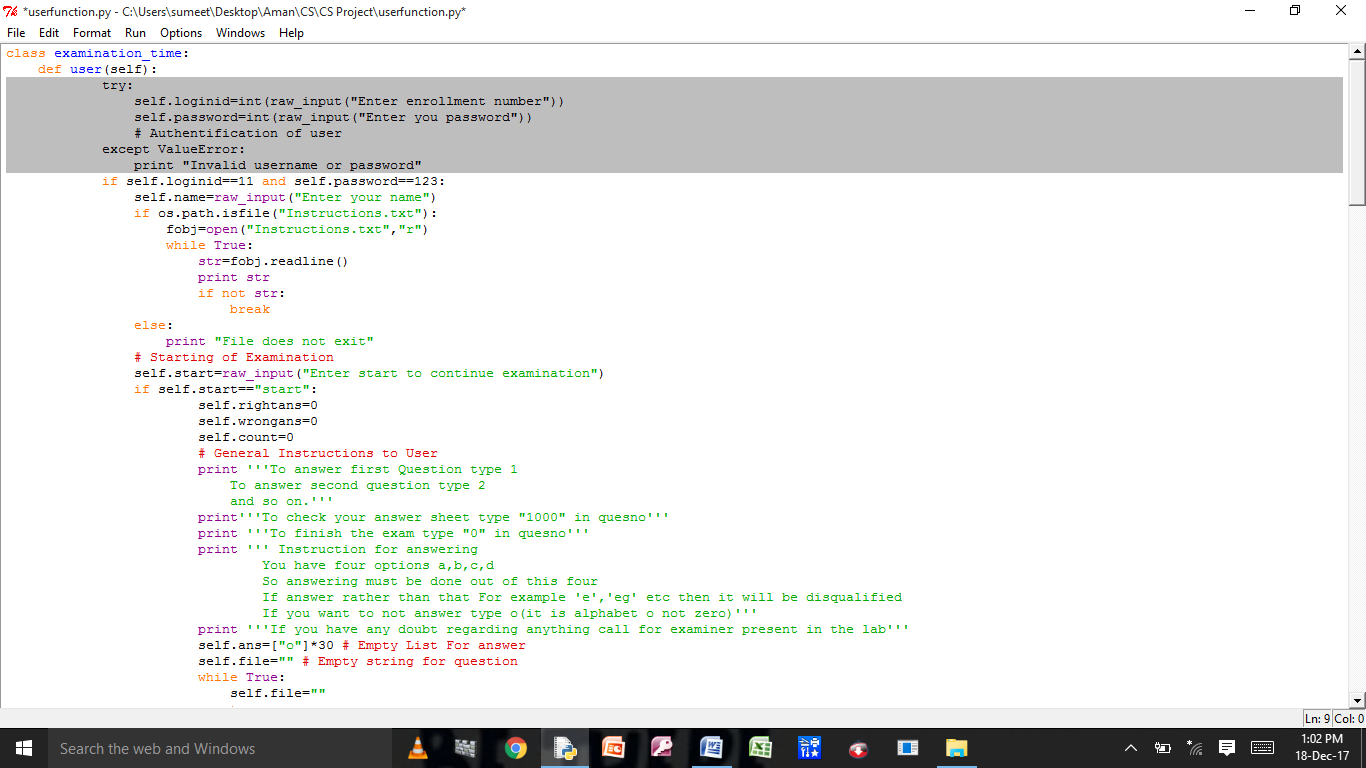
This one is the simple looping but the next one is the shine of the project.

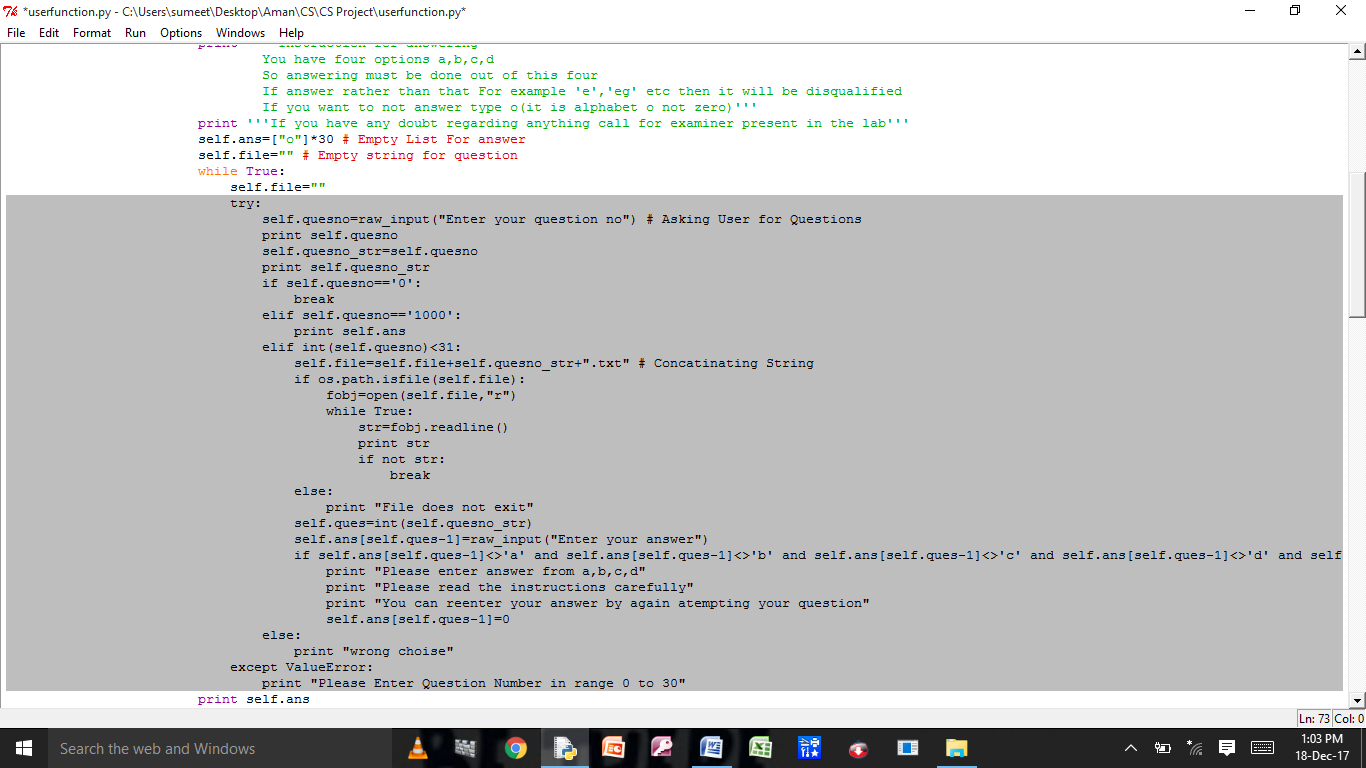


***IN A FUNCTION FOUR LOOP IS USED CONTINUSLY AND THIS IS THE THING WHICH I GET FROM MY PROJECT.***

Looping is used many times in my project and help me a lot.

* Exception Handling: The real meaning of exception handling is taught by my project. In the regular lectures’ when it is taught mostly the question arises of its use. But this project taught me its use in a very pleasant manner.





The second screenshot coding teaches me where to use try and except. Because if in this conding I have not use this feature then whenever a examiner gives values other than 30 a error occurs on the screen which stops my program and now this problem is solved by use of it.

-------------------------------------------------------------------------------------------------------------

**THINGS WHICH MY PROJECT TAUGHT TO ME**

* Firstly use of looping in a correct manner.

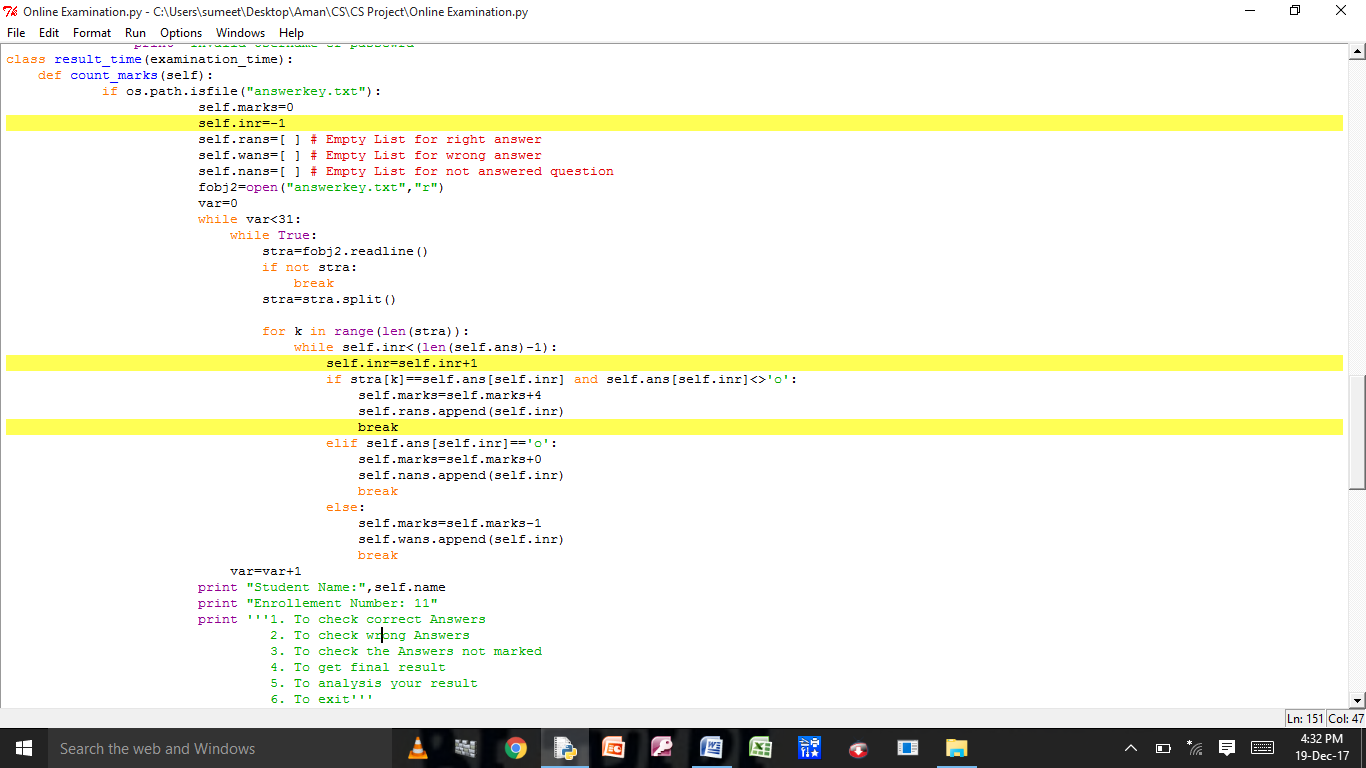
Already I have disused a lot regarding this topic. It seems to be very simple to use but the correct usage or I can say challenges came to know when we use it. Simple looping hardly tolerates us but when looping is done more than once or twice at a time it makes us cry. Same situation comes with me in this project.

The situation is that

- Taking answers from user

- Now comparing that answer with correct answer

- What’s the challenge: It is that correct answer is stored in file whereas user’s answer in a list. So if I apply loop for correct answer it makes hurdle in the path of user’s answer and same in the reverse situation. If I apply loop for both the answer if become the case of nested looping which means that a correct answer is compared with all the user’s answer and give false result. So after a lot of trials finally a solution has been came out for which *a big thanks to the concept of object oriented programming.* The solution is as follows.



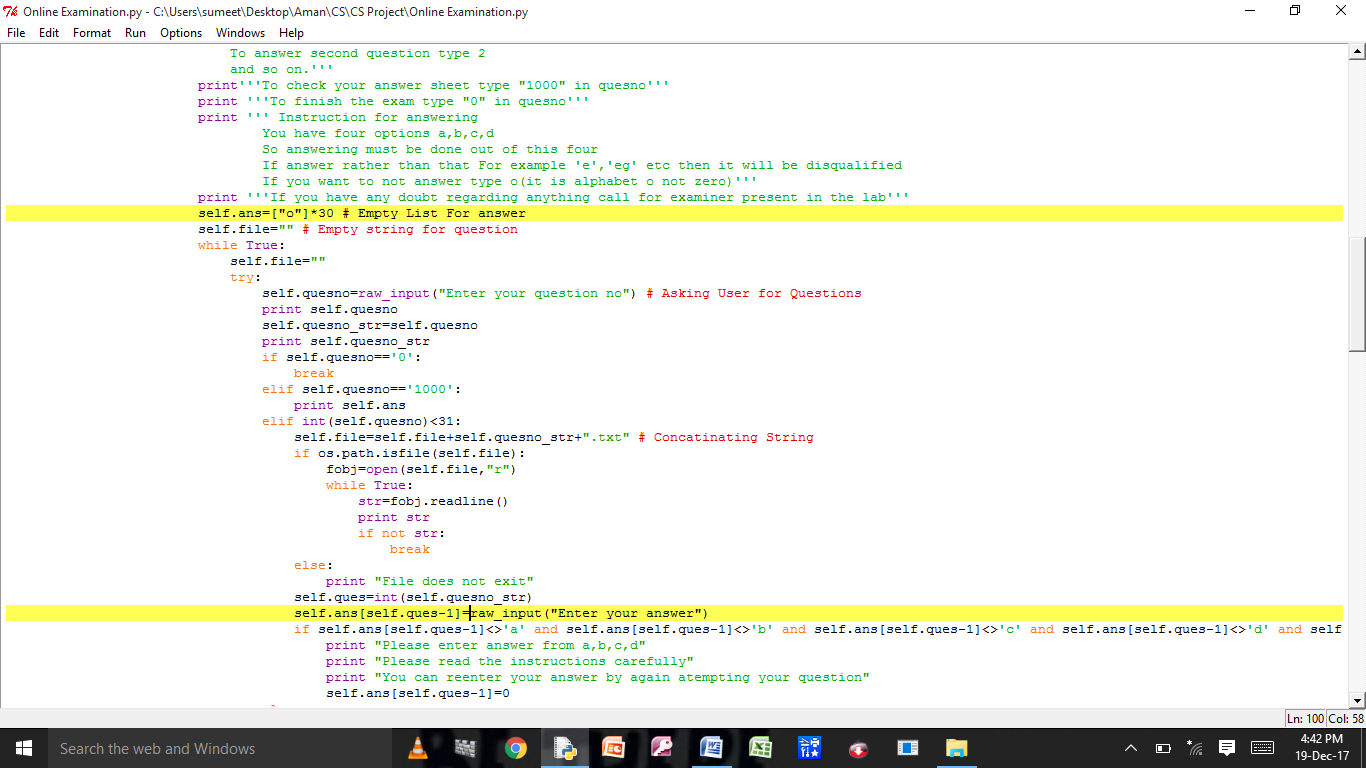
The solution is that a variable is taken with initial value as zero. And while executing it in the loop it doesn’t become a hurdle because I have use break statement there. Therefore my problem is solved and also taught me a lot.

* Second one is the multiple usage of list. List in class 11th seems to be just a variable to store many values. But now this feature of list is like a biggest tool for me in my project. When the situation come to take answer form user, list is used.

But what I learned mainly is that there is many ways to make a list. As we know that appending adds element at the end of the list, but my way is somehow different.

For example if a user gives the answer of question on 11, then the value of answer should be come on the 11th position of the list. And I am succeed in it with the simple concept of adding values in the list.

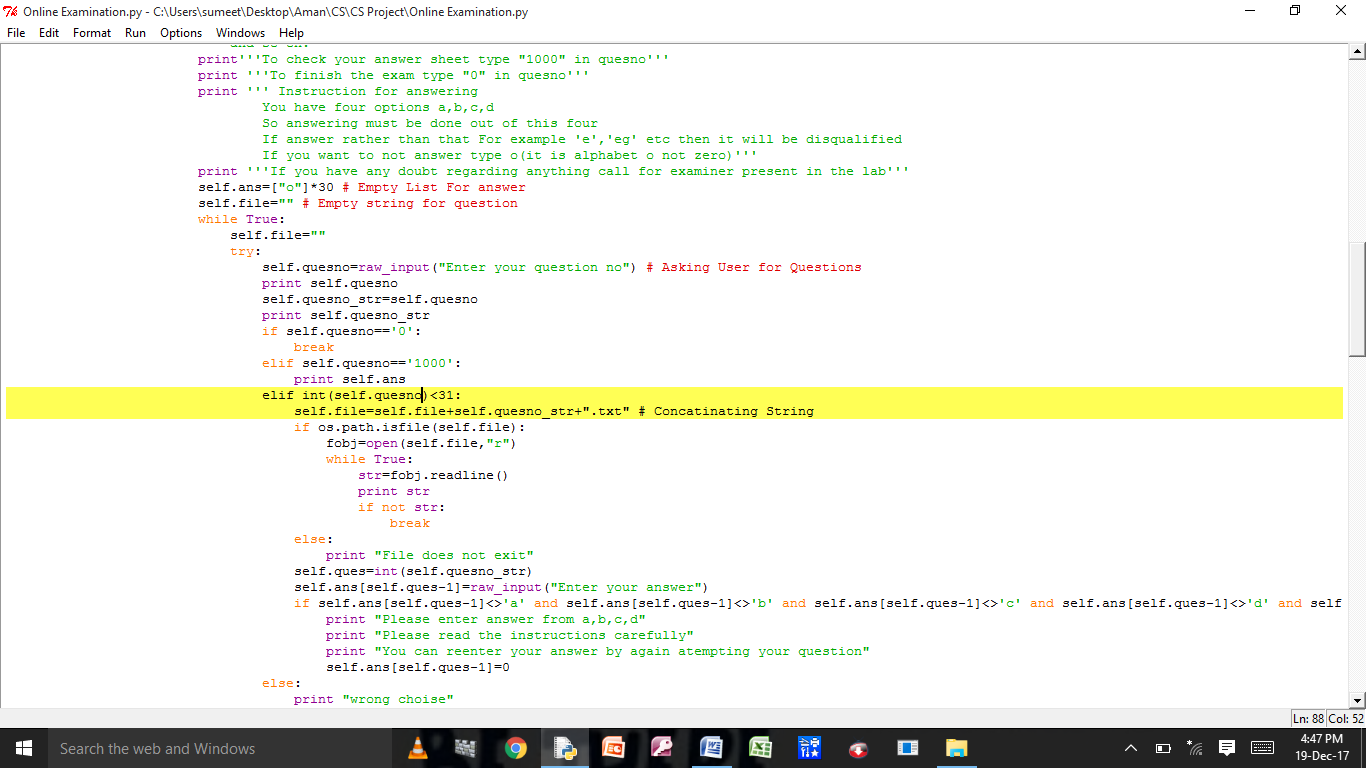
Besides that usual appending of values is also done for just distinguishing between the correct answer given by the user or wrong answer or questions that is not answered.



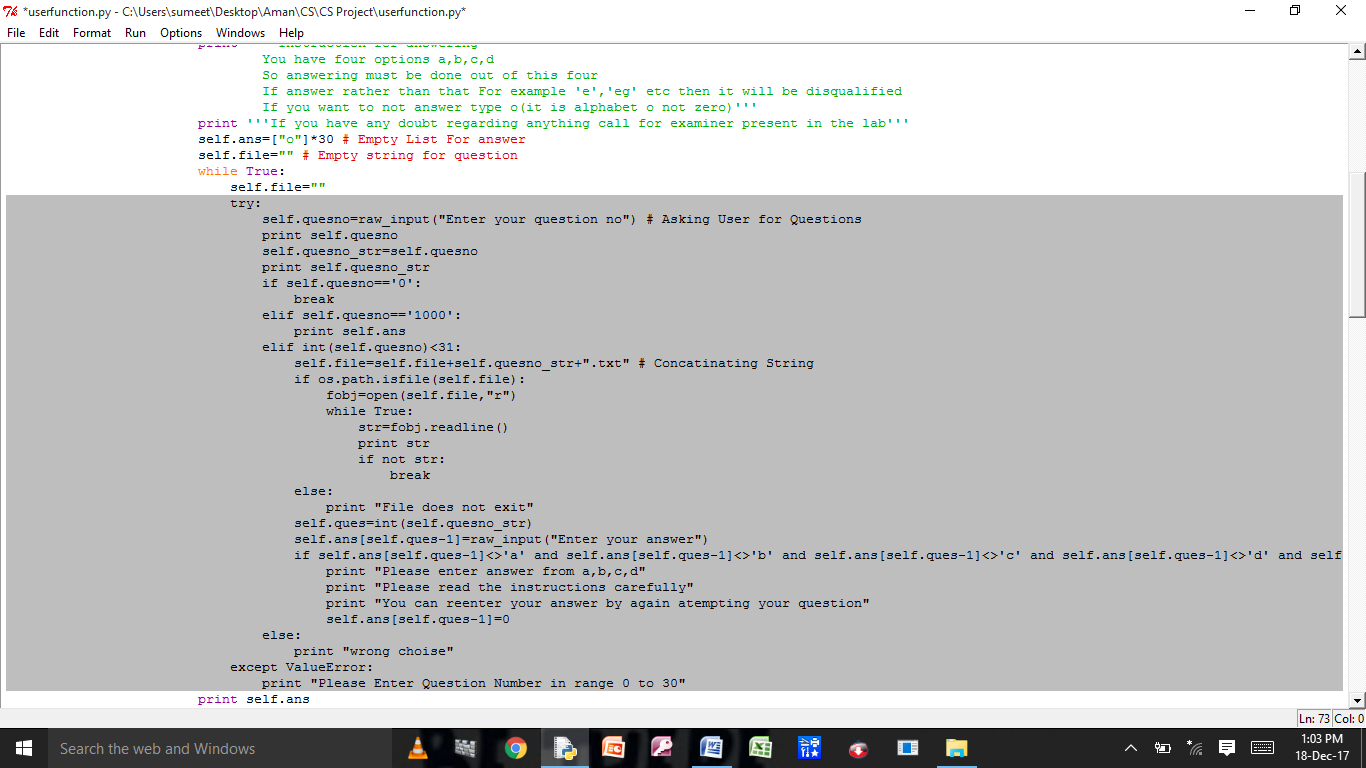
* Thirdly the usage of concatenating of string:

In calling the questions separately has also become challenge for me. By the concept of file handling either I can call all questions at a time or only one question. But I don’t want to do both, what I aim is that to call question one by one as user wants.

So a most important part of my project in which ***looping, file handing and concatenation is used together.*** That is something that I never thought can be done. So I can say that it is also very special for me form my project.



* Another concept is the exceptional handling in python. I can say that whatever I have early understood is only its application but its real meaning is understood in the project work only. As I already described that if an error generates in between the program it destroys whole of it, which is not required at that time. For example in my project only if user gives the question values as string rather than a integer by mistake then it is not so that my program to be stopped but it should give some message to the user and that what the exceptional handling concept works on.



* Last but not the least, the hold in the programming is improved somehow from my project. Programming work is not done that much in our regular classes mainly this year(i.e. in 12th standard) which somehow brings our confidence low. But after the completion of this project somehow from my side what I felt that at least I know the basic usage of many concept mostly which I disused earlier.

Somehow many programming error that has been created many times also gave basic idea where to improve.

So, overall what I feel that if this small project can taught me that much then I have lot to learn.

**-------------------------------------------------------------------------------------------------------------**

**SPECIALTIES OF MY PROJECT**

* The thought of building this project came to me when article of online exam of JEE(Joint Entrance Exam) from this year(i.e 2018) came in the newspaper. So, I also taught of making project on this topic. I have given online examination few times before making this exams.

Exam of KVPY and also few FIITJEE AITS test series is given by me which gave basic ideas to be done in my project. Few of them are that when user enters question number then only question comes on the screen which is not early in the mind of me. Next one is to give user a space that if he want to change the answer or want tot clear it he can.

Also he entered a answer and now want to make that answer clear than it is also done which is not provided early in my project.

* This is all about what I got ideas from many exams but few things which I feels that this should be in such exams. And I have out that in my project.

First thing is that instructions are very clear for the proper understanding to the person who is giving the exam.

Next is that it is very flexible for him to use. If he put some question no. or answer wrongly than also he is given change to change it.

* Since Graphics is not used in my project then also we have made it very clear for both the admin and the user to use it. Every step is very clear for both of them to use. Admin has also given many spaces in his work. He can easily change question number, he can easily change the answer which he has given earlier. So the point which I trying to tell that like other exams graphical approach is not given to him but flexibility is give.
* The main specialty of my project is the result declaration. After the completion of the exam only result is provided to the user in a very simple and expressive manner which I feels lack in other examination. We have provided different spaces for the questions which is answers wrongly or answer correctly or not answered. This overall performance is also judged at the result time.

So It is one of the thing of my project which make it different from others.

* Every programmer tires to give flexibility to the user and I also done it and disused a lot on it in the previous section. But my project differs from the other project in its coding. It is not that much complicated or I can say that not that much diverse that a programmer rather than me binds in it. It is straight forward pointing to the flowchart which I have made. I am writing this because whenever I asked doubt or can say queries regarding anything of project I have not to explain them all the things from the beginning.
* Something what I feel that after making this project or while giving this brief introduction about my project now, I feels that YES the work which I have done is succeed. Many things I can still do on my project, but till now what I made is worth remembering experience for me to be remembered.

Use of the comment is also there for the better understanding in the project.

So this is my first project in this field, which taught me a lot(already disused), and I can say it is specialty in itself only.

-------------------------------------------------------------------------------------------------

***Project Coding***

*# My Python Program "Online Examination"*

*import os*

*class online\_examination:*

*# Admin Function*

*def admin(self):*

*self.Loginid=int(raw\_input("Enter your seven digit username"))*

*self.password=raw\_input("Enter your password")*

*#Authentification of admin*

*if self.Loginid==1234567 and self.password=="password":*

*print '''To read the program file of question type "1"*

*To append(change) the program file of question type "2"*

*To read the answer of the choosen question type "3"*

*To Logout successfully type "4"'''*

*# Execution of user function*

*# Question program file must be created before execution*

*while True:*

*b=int(raw\_input("Enter your appropiate choise from above"))*

*if b==1:*

*if os.path.isfile("Questions program file.txt"):*

*fobj=open("Questions program file.txt","r")*

*while True:*

*str=fobj.readline()*

*print str*

*if not str:*

*break*

*else:*

*print "File does not exit"*

*if b==2:*

*print "If you want to change the data of the file,go to the source file and edit it"*

*if b==3:*

*if os.path.isfile("answerkey.txt"):*

*fobj=open("answerkey.txt","r")*

*while True:*

*str=fobj.readline()*

*print str*

*if not str:*

*break*

*if b==4:*

*print "Successfully logout"*

*break*

*# Successfully Logout from admin function*

*else:*

*print "Invalid username or password"*

*class examination\_time:*

*def user(self):*

*try:*

*self.loginid=int(raw\_input("Enter enrollment number"))*

*self.password=int(raw\_input("Enter you password"))*

*# Authentification of user*

*except ValueError:*

*print "Invalid username or password"*

*if self.loginid==11 and self.password==123:*

*self.name=raw\_input("Enter your name")*

*if os.path.isfile("Instructions.txt"):*

*fobj=open("Instructions.txt","r")*

*while True:*

*str=fobj.readline()*

*print str*

*if not str:*

*break*

*else:*

*print "File does not exit"*

*# Starting of Examination*

*self.start=raw\_input("Enter start to continue examination")*

*if self.start=="start":*

*self.rightans=0*

*self.wrongans=0*

*self.count=0*

*# General Instructions to User*

*print '''To answer first Question type 1*

*To answer second question type 2*

*and so on.'''*

*print'''To check your answer sheet type "1000" in quesno'''*

*print '''To finish the exam type "0" in quesno'''*

*print ''' Instruction for answering*

*You have four options a,b,c,d*

*So answering must be done out of this four*

*If answer rather than that For example 'e','eg' etc then it will be disqualified*

*If you want to not answer type o(it is alphabet o not zero)'''*

*print '''If you have any doubt regarding anything call for examiner present in the lab'''*

*self.ans=["o"]\*30 # Empty List For answer*

*self.file="" # Empty string for question*

*while True:*

*self.file=""*

*try:*

*self.quesno=raw\_input("Enter your question no") # Asking User for Questions*

*print self.quesno*

*self.quesno\_str=self.quesno*

*print self.quesno\_str*

*if self.quesno=='0':*

*break*

*elif self.quesno=='1000':*

*print self.ans*

*elif int(self.quesno)<31:*

*self.file=self.file+self.quesno\_str+".txt" # Concatinating String*

*if os.path.isfile(self.file):*

*fobj=open(self.file,"r")*

*while True:*

*str=fobj.readline()*

*print str*

*if not str:*

*break*

*else:*

*print "File does not exit"*

*self.ques=int(self.quesno\_str)*

*self.ans[self.ques-1]=raw\_input("Enter your answer")*

*if self.ans[self.ques-1]<>'a' and self.ans[self.ques-1]<>'b' and self.ans[self.ques-1]<>'c' and self.ans[self.ques-1]<>'d' and self.ans[self.ques-1]<>'o':*

*print "Please enter answer from a,b,c,d"*

*print "Please read the instructions carefully"*

*print "You can reenter your answer by again atempting your question"*

*self.ans[self.ques-1]=0*

*else:*

*print "wrong choise"*

*except ValueError:*

*print "Please Enter Question Number in range 0 to 30"*

*print self.ans*

*else:*

*print "You have not started your exam"*

*else:*

*print "Invalid Username or passowrd"*

*class result\_time(examination\_time):*

*def count\_marks(self):*

*if os.path.isfile("answerkey.txt"):*

*self.marks=0*

*self.inr=-1*

*self.rans=[ ] # Empty List for right answer*

*self.wans=[ ] # Empty List for wrong answer*

*self.nans=[ ] # Empty List for not answered question*

*fobj2=open("answerkey.txt","r")*

*var=0*

*while var<31:*

*while True:*

*stra=fobj2.readline()*

*if not stra:*

*break*

*stra=stra.split()*

*for k in range(len(stra)):*

*while self.inr<(len(self.ans)-1):*

*self.inr=self.inr+1*

*if stra[k]==self.ans[self.inr] and self.ans[self.inr]<>'o':*

*self.marks=self.marks+4*

*self.rans.append(self.inr)*

*break*

*elif self.ans[self.inr]=='o':*

*self.marks=self.marks+0*

*self.nans.append(self.inr)*

*break*

*else:*

*self.marks=self.marks-1*

*self.wans.append(self.inr)*

*break*

*var=var+1*

*print "Student Name:",self.name*

*print "Enrollement Number: 11"*

*print '''1. To check correct Answers*

*2. To check wrong Answers*

*3. To check the Answers not marked*

*4. To get final result*

*5. To analysis your result*

*6. To exit'''*

*while True:*

*y=int(raw\_input("Enter your choise"))*

*if y==1:*

*if len(self.rans)==0:*

*print "No question is answered correctly"*

*else:*

*for r in range(len(self.rans)):*

*print "Question:",self.rans[r]+1 #Calculating right answer*

*print "Your answered",self.ans[self.rans[r]]*

*print "You answered correctly"*

*elif y==2:*

*if len(self.wans)==0:*

*print "No question is answered wrongly"*

*else:*

*for w in range(len(self.wans)):*

*print "Question:",self.wans[w]+1 #Calculating wrong answer*

*print "Your answered",self.ans[self.wans[w]]*

*print "You answered wrongly"*

*elif y==3:*

*if len(self.nans)==0:*

*print "Every question is marked"*

*else:*

*for n in range(len(self.nans)):*

*print "Question:",self.nans[n]+1*

*print "Your answered",self.ans[self.nans[n]]*

*print "You did not answered"*

*elif y==4:*

*print "Total marks",self.marks*

*elif y==5:*

*if self.marks<=10:*

*print "Below average performance"*

*print "You can do more well"*

*if self.marks>10 and self.marks<=25:*

*print "Average performance"*

*print "You can do more well and you will"*

*if self.marks>25 and self.marks<=50:*

*print "Fair Performance"*

*print "You can do more well and you will"*

*if self.marks>50 and self.marks<=75:*

*print "Good performance"*

*print "You can do more well and you will"*

*if self.marks>75 and self.marks<=100:*

*print "Very Good performance"*

*print "Now Go for a full score"*

*if self.marks>100 and self.marks<=115:*

*print "Excellent performance"*

*print "Now its turn for full score"*

*if self.marks>115 and self.marks<=120:*

*print "Extraordinary Performance"*

*print "You are the star of this test"*

*print "All the best for your upcoming exams and keep it up like this"*

*else:*

*break*

*else:*

*print "Answerkey answerkey does not exist"*

*print '''If you are admin type "1"*

*If you are user type "2" '''*

*#Asking User for his appropiate Choise*

*a=int(raw\_input("Enter your appropiate choise from above selection"))*

*if a==1:*

*obj=online\_examination()*

*obj.admin()*

*elif a==2:*

*obj=result\_time()*

*obj.user()*

*obj.count\_marks()*

*else:*

*"Please Choose appropiate choise from above"*

Admin Function

import os

print '''To read the program file of question type "1"

To append(change) the program file of question type "2"

To read the answer of the choosen question type "3"

To Logout successfully type "4"'''

# Execution of user function

# Question program file must be created before execution

while True:

b=int(raw\_input("Enter your appropiate choise from above"))

if b==1:

if os.path.isfile("Questions program file.txt"):

fobj=open("Questions program file.txt","r")

while True:

str=fobj.readline()

print str

if not str:

break

else:

print "File does not exit"

if b==2:

print "If you want to change the data of the file,go to the source file and edit it"

if b==3:

if os.path.isfile("answerkey.txt"):

fobj=open("answerkey.txt","r")

while True:

str=fobj.readline()

print str

if not str:

break

if b==4:

print "Successfully logout"

break

User Function

class examination\_time:

def user(self):

try:

self.loginid=int(raw\_input("Enter enrollment number"))

self.password=int(raw\_input("Enter you password"))

# Authentification of user

except ValueError:

print "Invalid username or password"

if self.loginid==11 and self.password==123:

self.name=raw\_input("Enter your name")

if os.path.isfile("Instructions.txt"):

fobj=open("Instructions.txt","r")

while True:

str=fobj.readline()

print str

if not str:

break

else:

print "File does not exit"

# Starting of Examination

self.start=raw\_input("Enter start to continue examination")

if self.start=="start":

self.rightans=0

self.wrongans=0

self.count=0

# General Instructions to User

print '''To answer first Question type 1

To answer second question type 2

and so on.'''

print'''To check your answer sheet type "1000" in quesno'''

print '''To finish the exam type "0" in quesno'''

print ''' Instruction for answering

You have four options a,b,c,d

So answering must be done out of this four

If answer rather than that For example 'e','eg' etc then it will be disqualified

If you want to not answer type o(it is alphabet o not zero)'''

print '''If you have any doubt regarding anything call for examiner present in the lab'''

self.ans=["o"]\*30 # Empty List For answer

self.file="" # Empty string for question

while True:

self.file=""

try:

self.quesno=raw\_input("Enter your question no") # Asking User for Questions

print self.quesno

self.quesno\_str=self.quesno

print self.quesno\_str

if self.quesno=='0':

break

elif self.quesno=='1000':

print self.ans

elif int(self.quesno)<31:

self.file=self.file+self.quesno\_str+".txt" # Concatinating String

if os.path.isfile(self.file):

fobj=open(self.file,"r")

while True:

str=fobj.readline()

print str

if not str:

break

else:

print "File does not exit"

self.ques=int(self.quesno\_str)

self.ans[self.ques-1]=raw\_input("Enter your answer")

if self.ans[self.ques-1]<>'a' and self.ans[self.ques-1]<>'b' and self.ans[self.ques-1]<>'c' and self.ans[self.ques-1]<>'d' and self.ans[self.ques-1]<>'o':

print "Please enter answer from a,b,c,d"

print "Please read the instructions carefully"

print "You can reenter your answer by again atempting your question"

self.ans[self.ques-1]=0

else:

print "wrong choise"

except ValueError:

print "Please Enter Question Number in range 0 to 30"

print self.ans

else:

print "You have not started your exam"

else:

print "Invalid Username or passowrd"

class result\_time(examination\_time):

def count\_marks(self):

if os.path.isfile("answerkey.txt"):

self.marks=0

self.inr=-1

self.rans=[ ] # Empty List for right answer

self.wans=[ ] # Empty List for wrong answer

self.nans=[ ] # Empty List for not answered question

fobj2=open("answerkey.txt","r")

var=0

while var<31:

while True:

stra=fobj2.readline()

if not stra:

break

stra=stra.split()

for k in range(len(stra)):

while self.inr<(len(self.ans)-1):

self.inr=self.inr+1

if stra[k]==self.ans[self.inr] and self.ans[self.inr]<>'o':

self.marks=self.marks+4

self.rans.append(self.inr)

break

elif self.ans[self.inr]=='o':

self.marks=self.marks+0

self.nans.append(self.inr)

break

else:

self.marks=self.marks-1

self.wans.append(self.inr)

break

var=var+1

print "Student Name:",self.name

print "Enrollement Number: 11"

print '''1. To check correct Answers

2. To check wrong Answers

3. To check the Answers not marked

4. To get final result

5. To analysis your result

6. To exit'''

while True:

y=int(raw\_input("Enter your choise"))

if y==1:

if len(self.rans)==0:

print "No question is answered correctly"

else:

for r in range(len(self.rans)):

print "Question:",self.rans[r]+1 #Calculating right answer

print "Your answered",self.ans[self.rans[r]]

print "You answered correctly"

elif y==2:

if len(self.wans)==0:

print "No question is answered wrongly"

else:

for w in range(len(self.wans)):

print "Question:",self.wans[w]+1 #Calculating wrong answer

print "Your answered",self.ans[self.wans[w]]

print "You answered wrongly"

elif y==3:

if len(self.nans)==0:

print "Every question is marked"

else:

for n in range(len(self.nans)):

print "Question:",self.nans[n]+1

print "Your answered",self.ans[self.nans[n]]

print "You did not answered"

elif y==4:

print "Total marks",self.marks

elif y==5:

if self.marks<=10:

print "Below average performance"

print "You can do more well"

if self.marks>10 and self.marks<=25:

print "Average performance"

print "You can do more well and you will"

if self.marks>25 and self.marks<=50:

print "Fair Performance"

print "You can do more well and you will"

if self.marks>50 and self.marks<=75:

print "Good performance"

print "You can do more well and you will"

if self.marks>75 and self.marks<=100:

print "Very Good performance"

print "Now Go for a full score"

if self.marks>100 and self.marks<=115:

print "Excellent performance"

print "Now its turn for full score"

if self.marks>115 and self.marks<=120:

print "Extraordinary Performance"

print "You are the star of this test"

print "All the best for your upcoming exams and keep it up like this"

else:

break

else:

print "Answerkey answerkey does not exist"

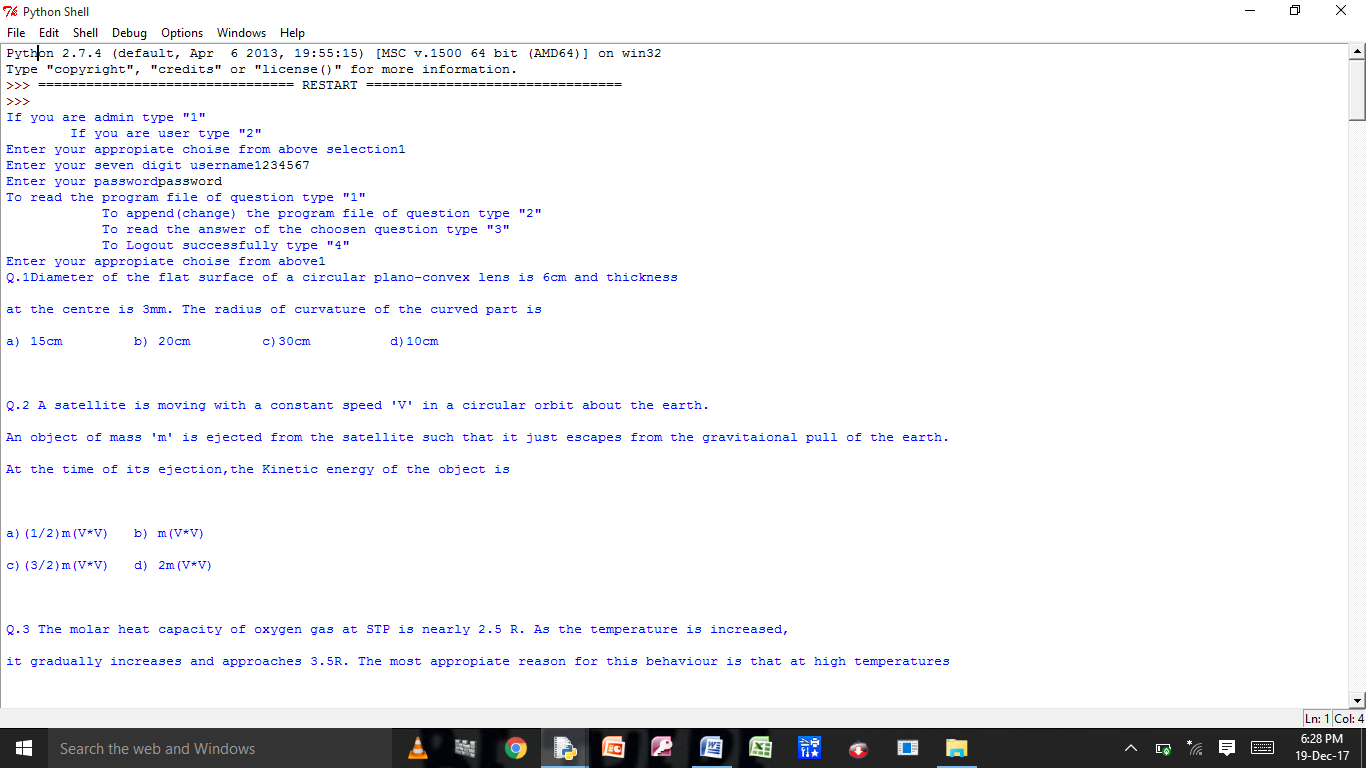
obj=result\_time()

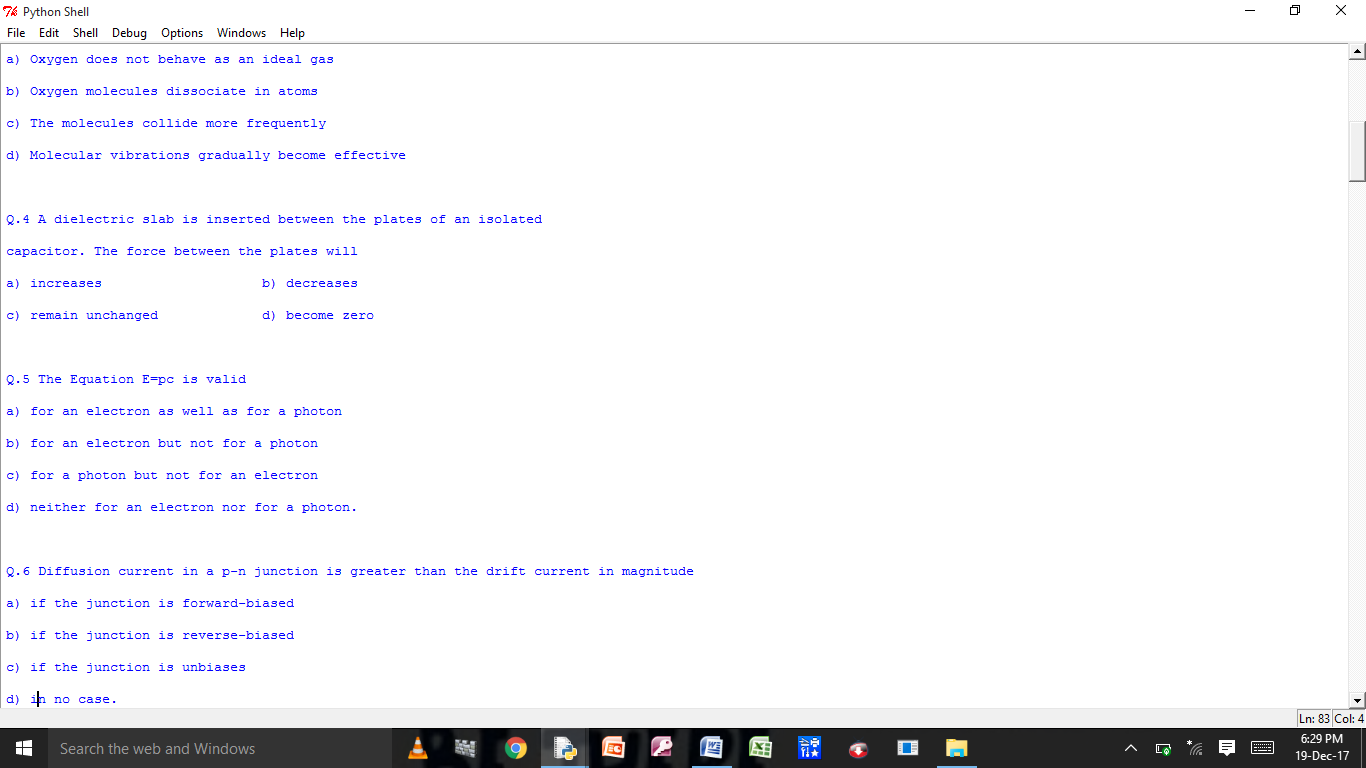
obj.user()

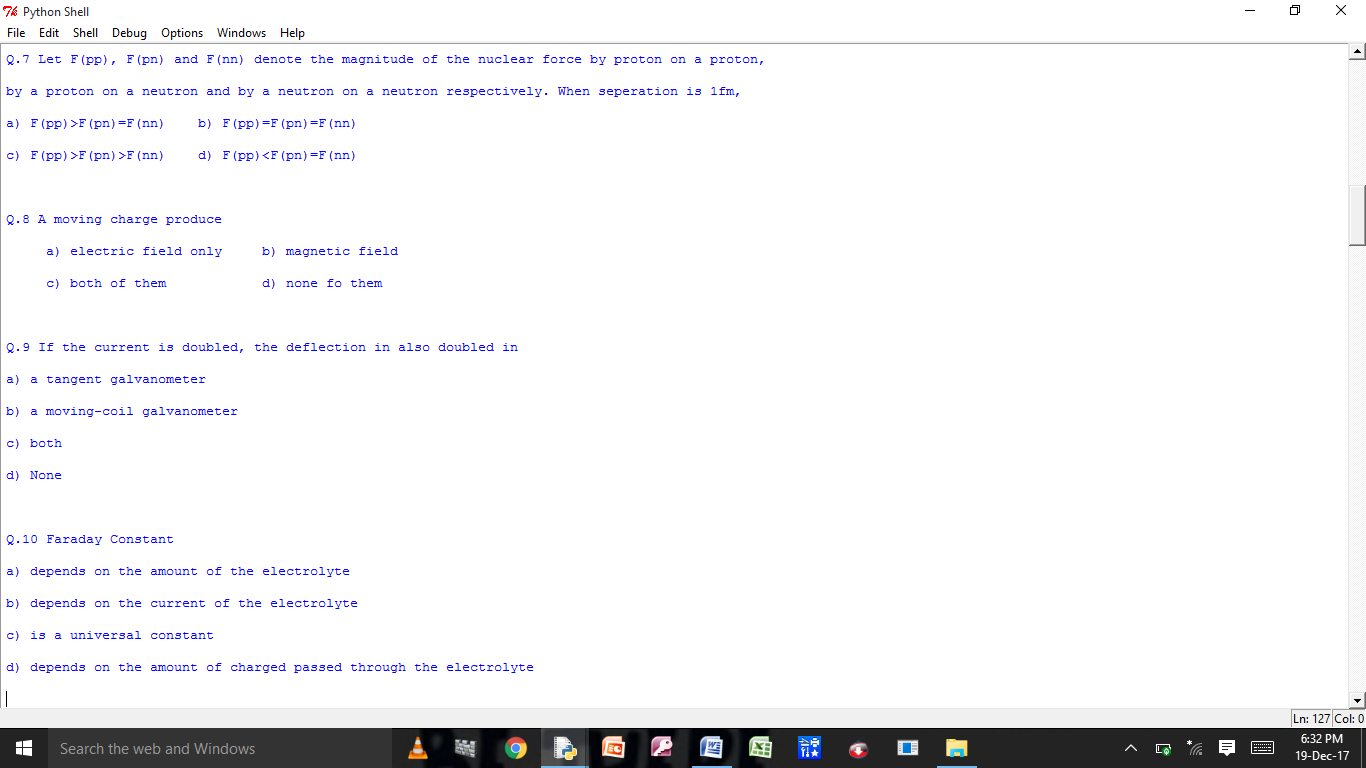
obj.count\_marks()

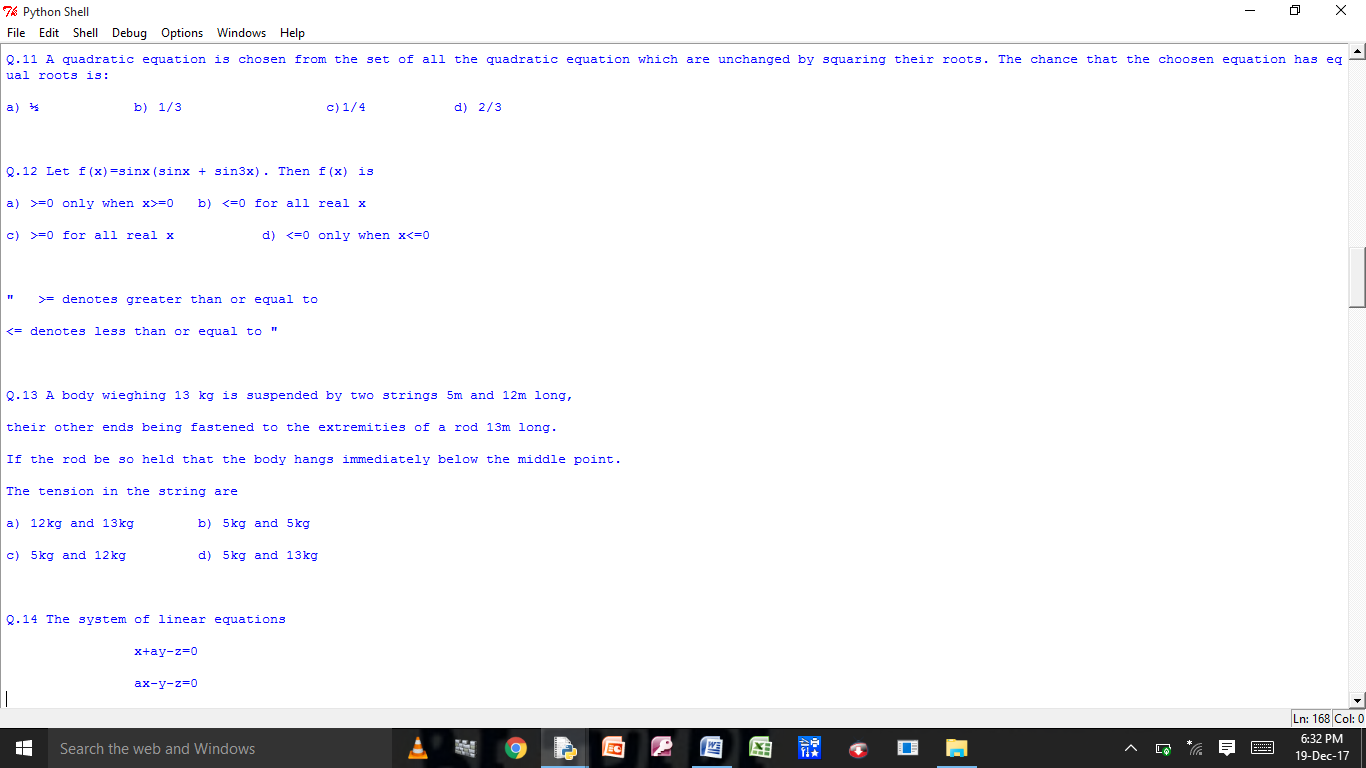
----------------------------------------------------------------------------------------------------------------

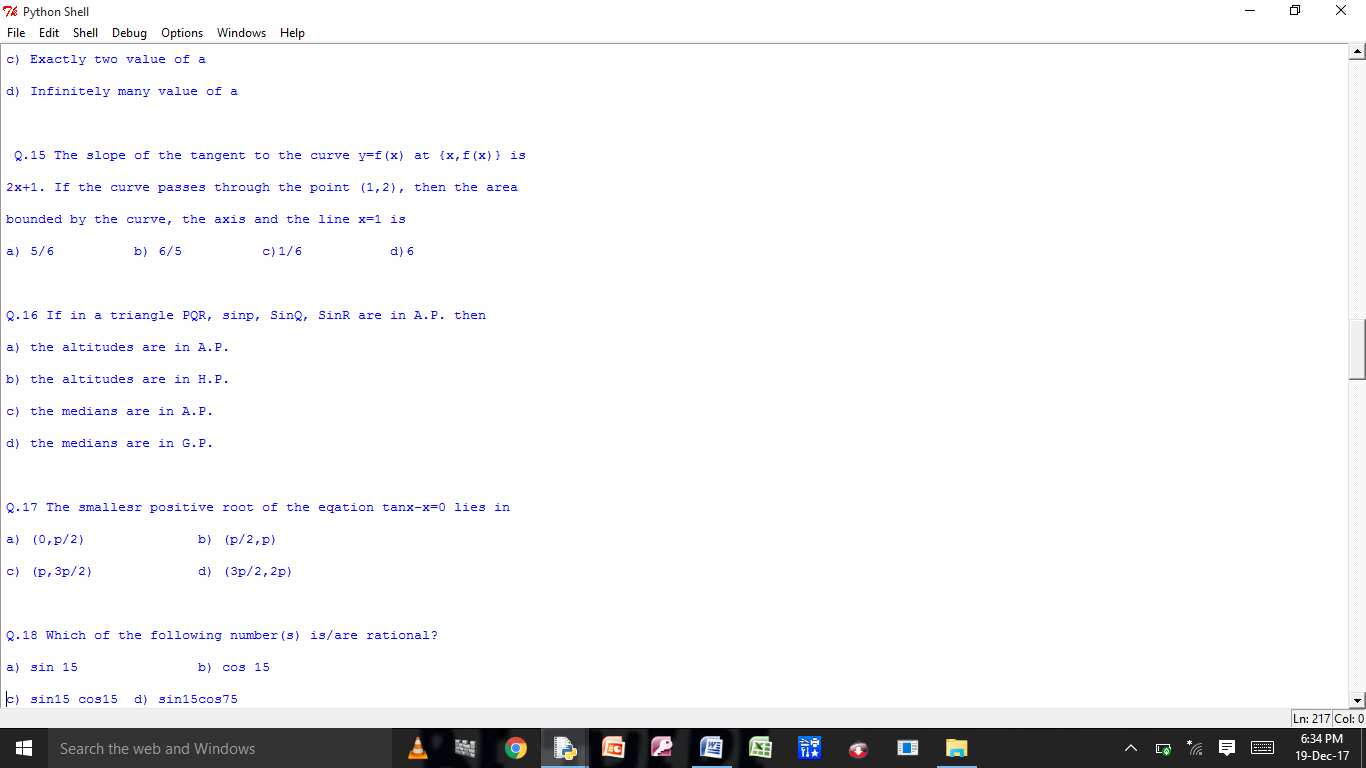
***Execution of Coding***

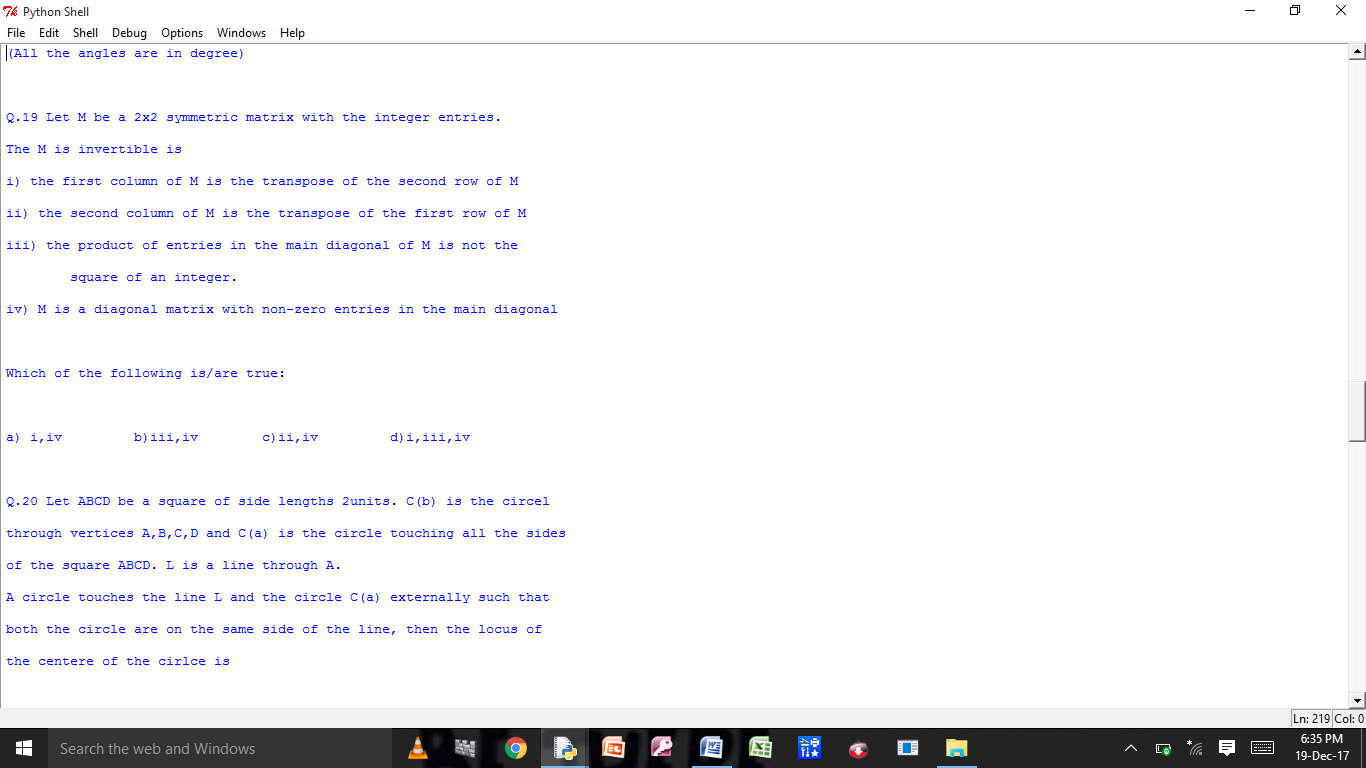


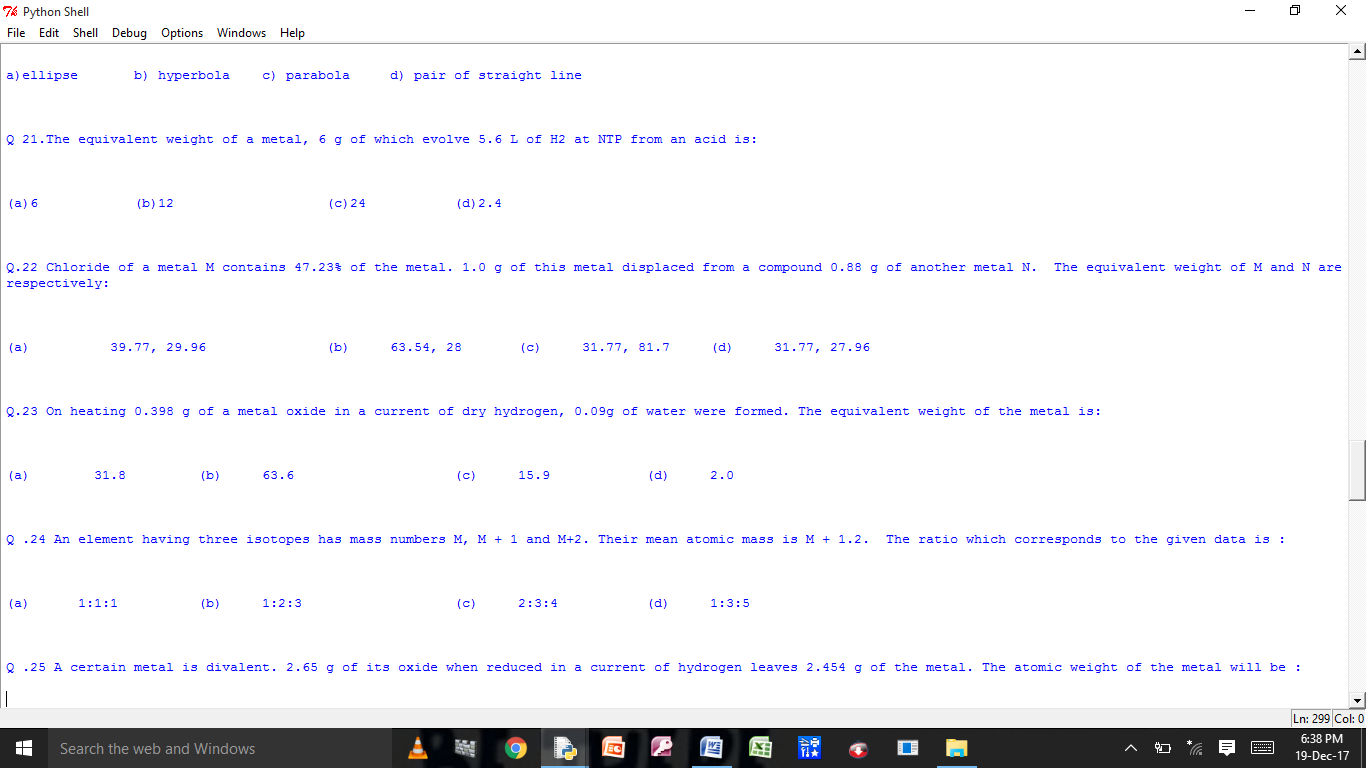


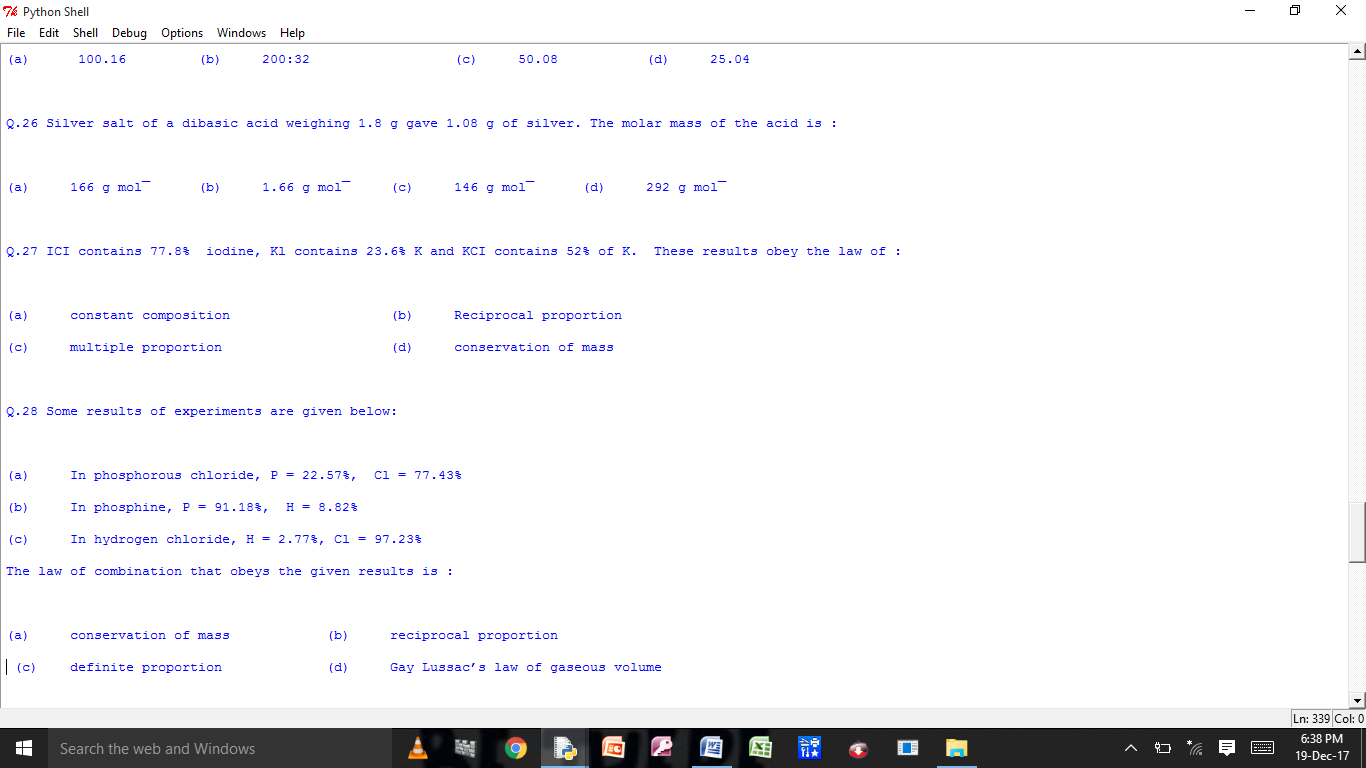


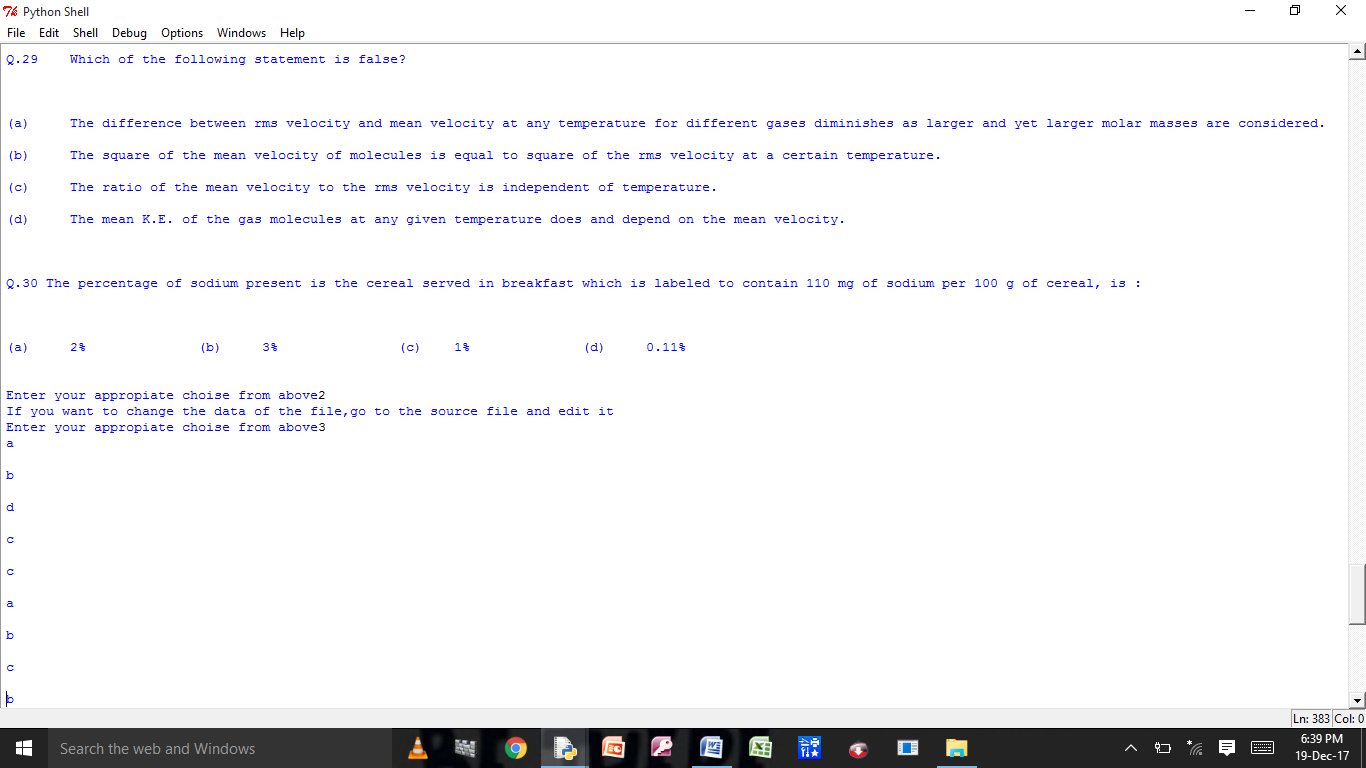


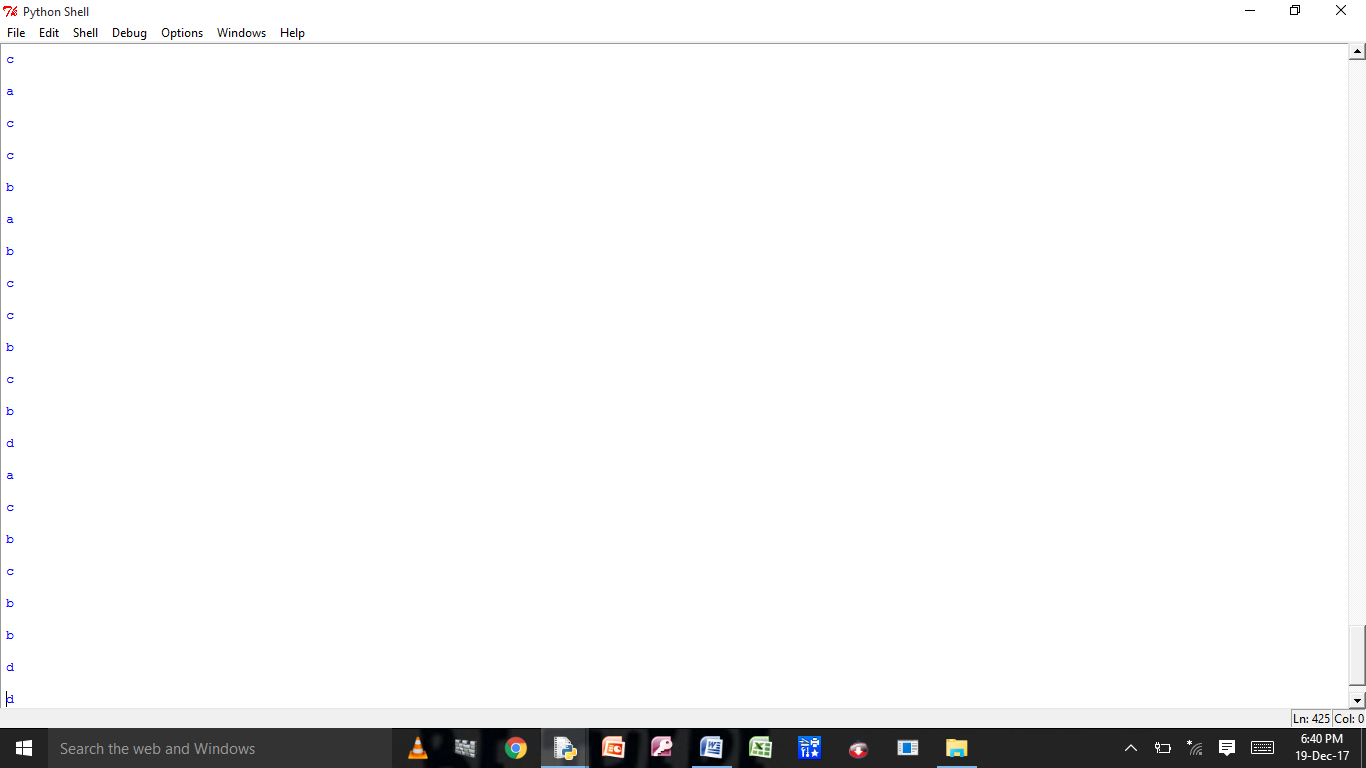


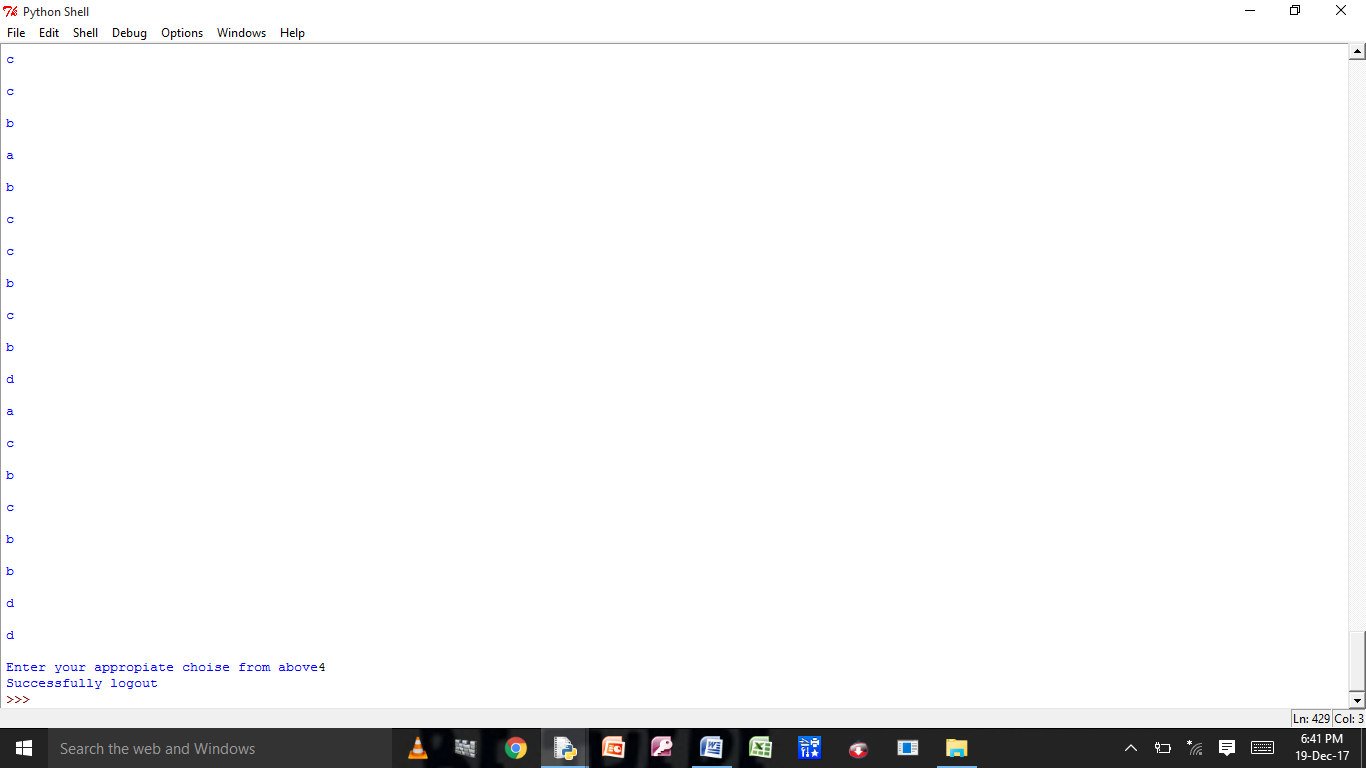




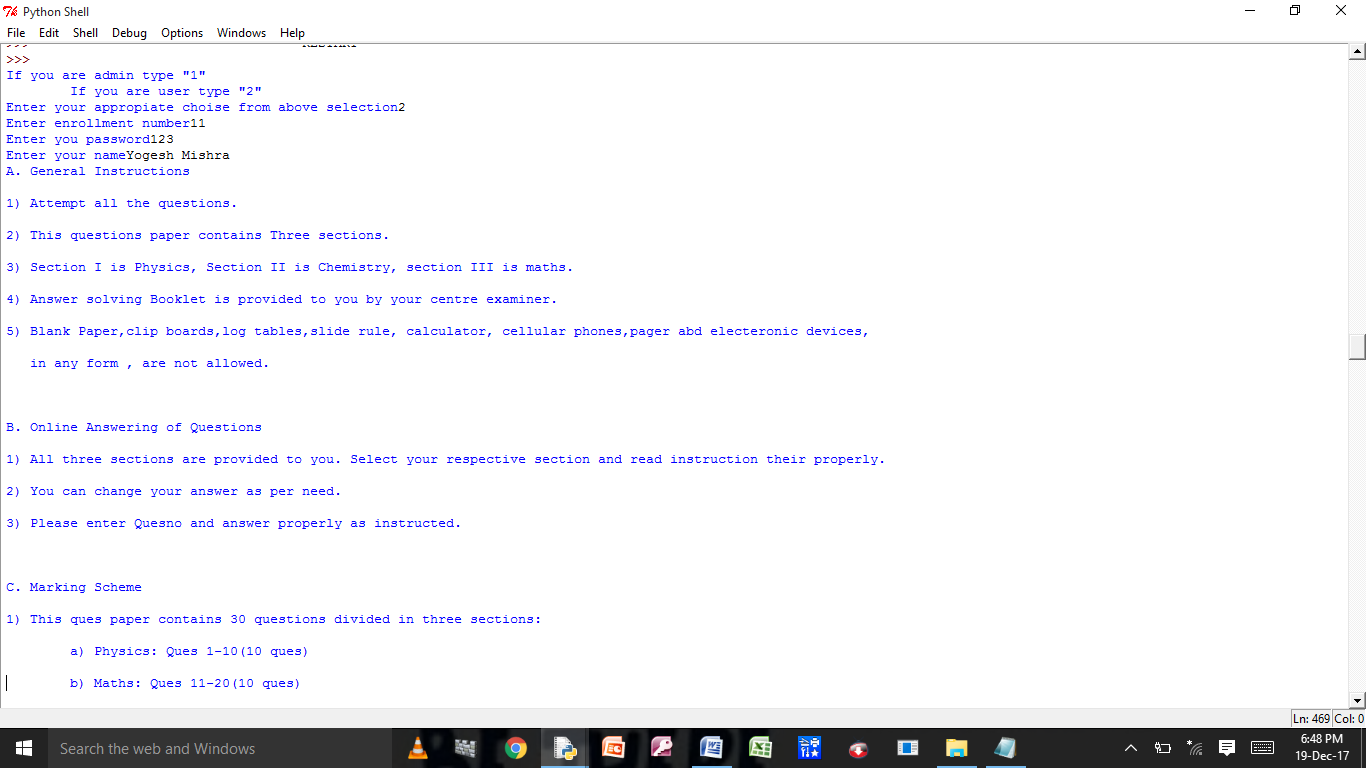


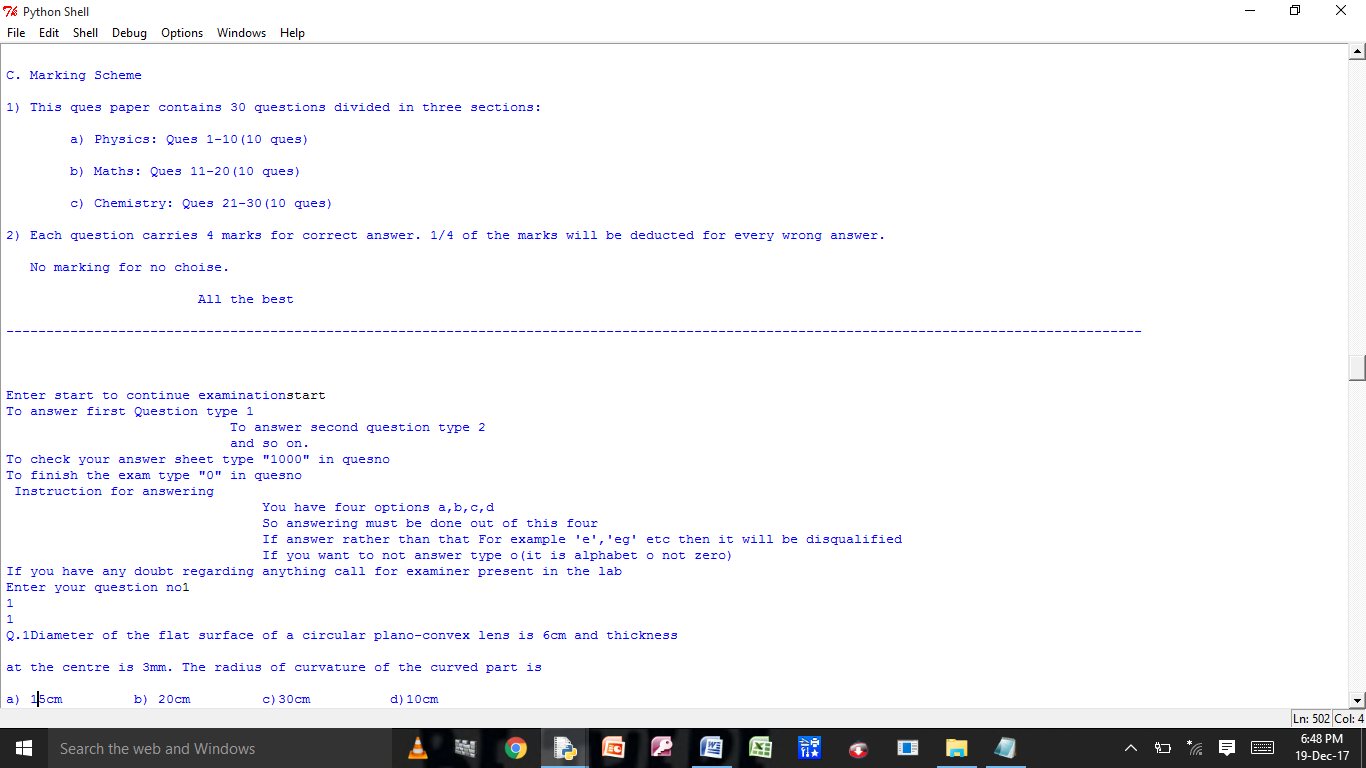


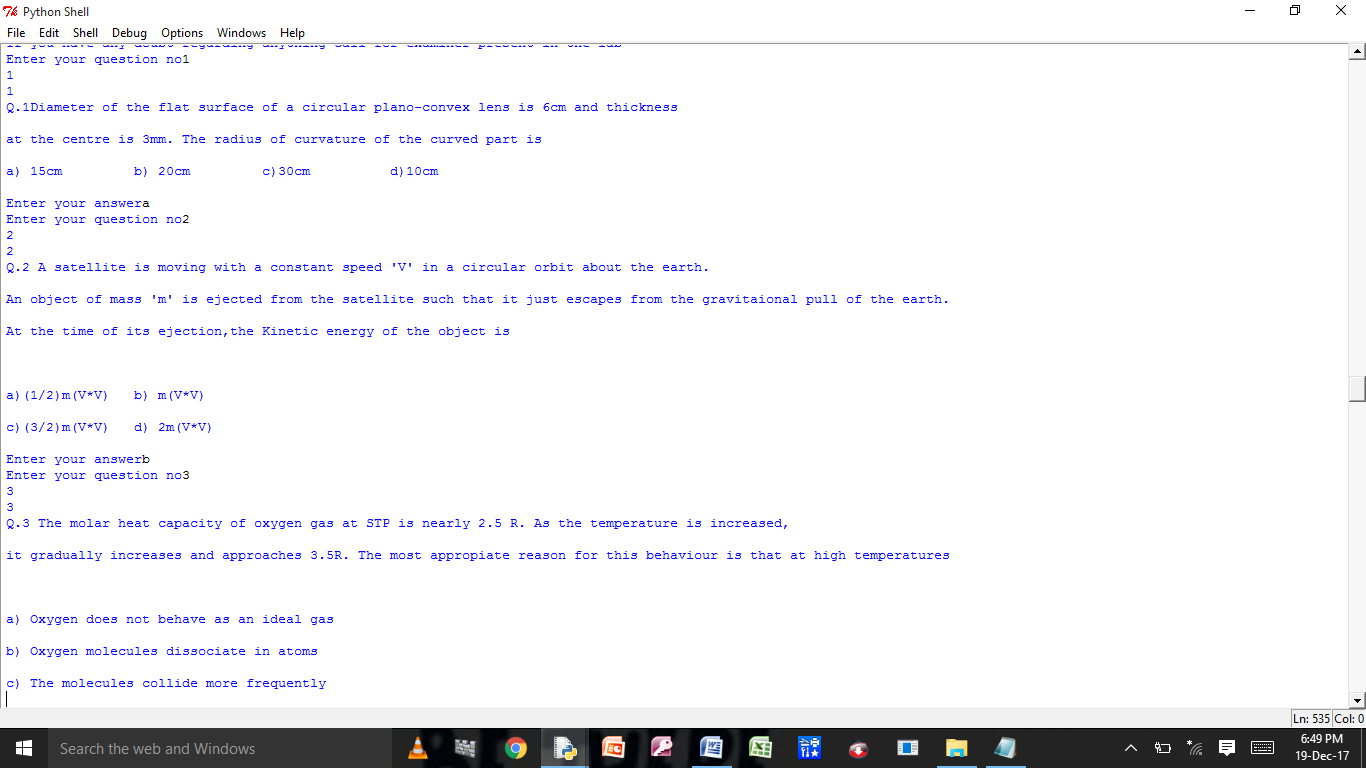


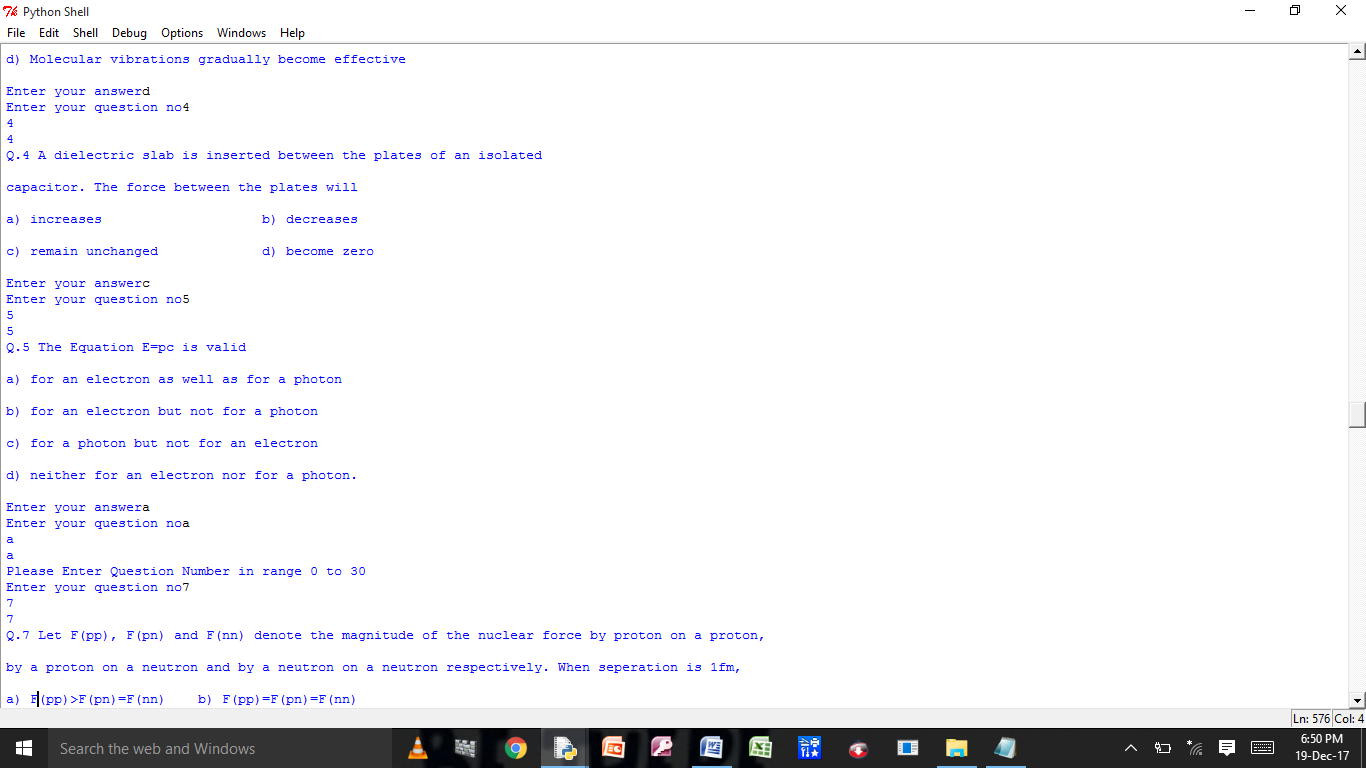


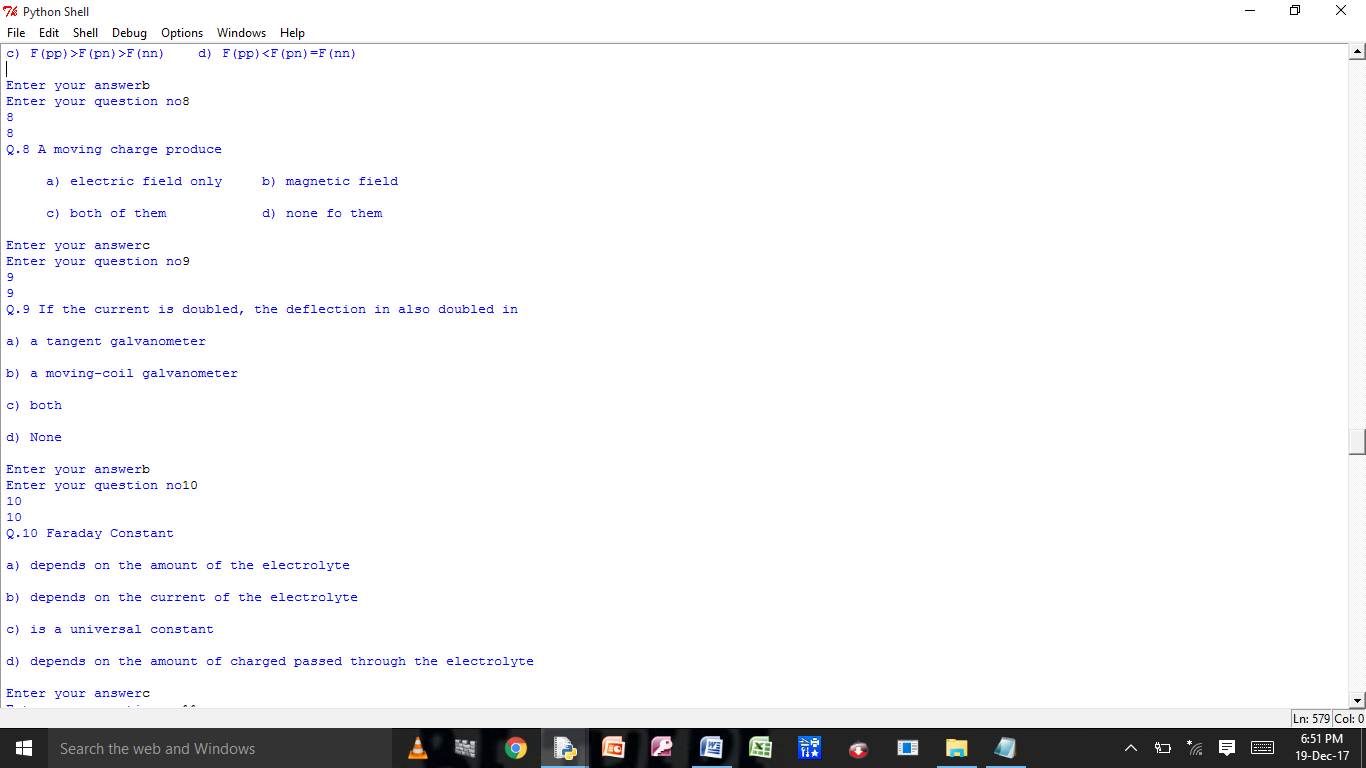
For User

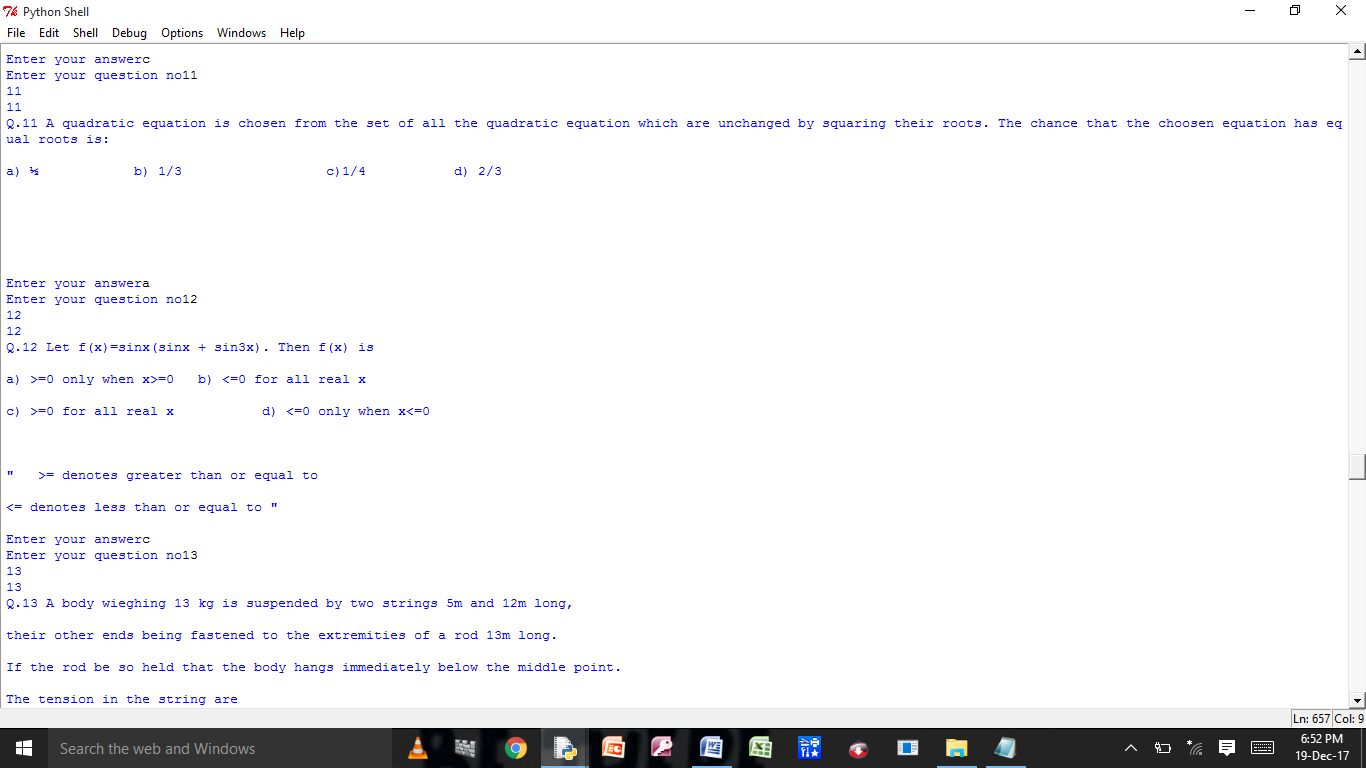


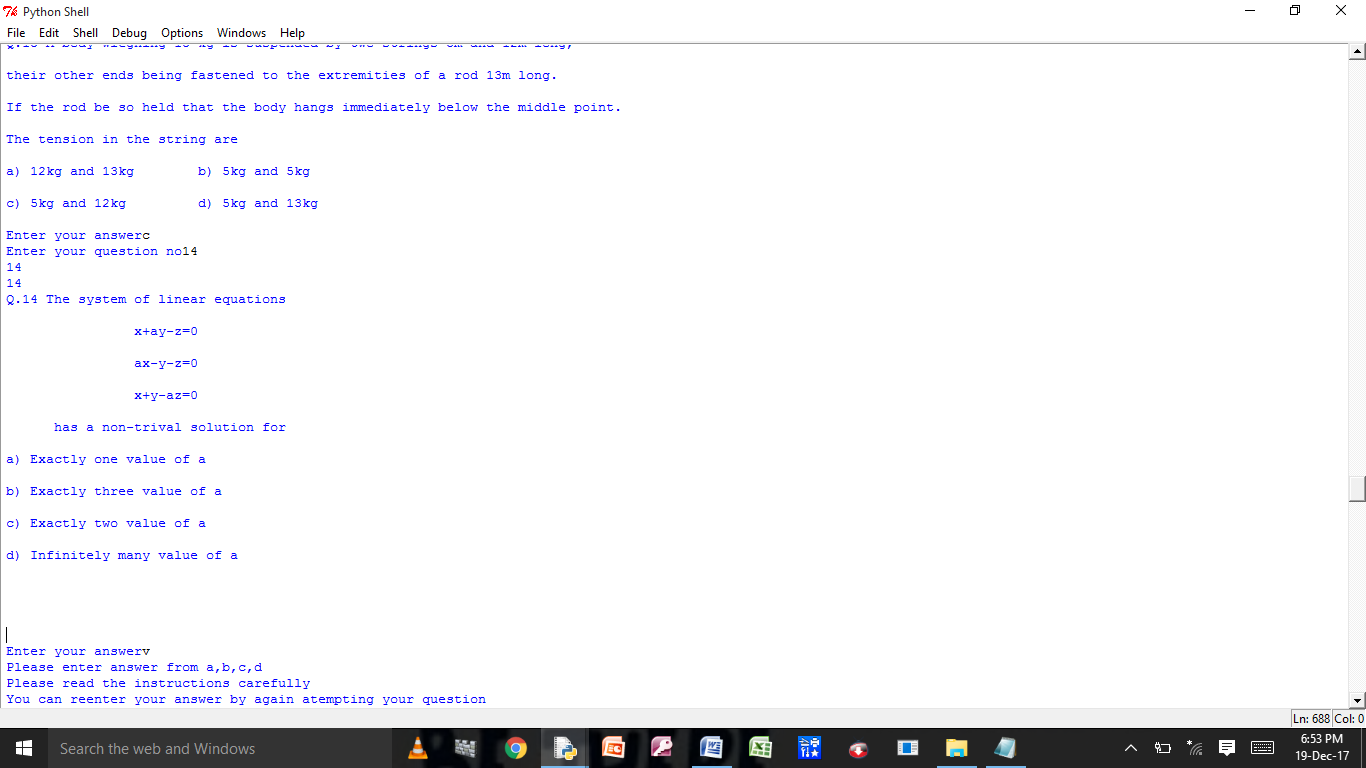


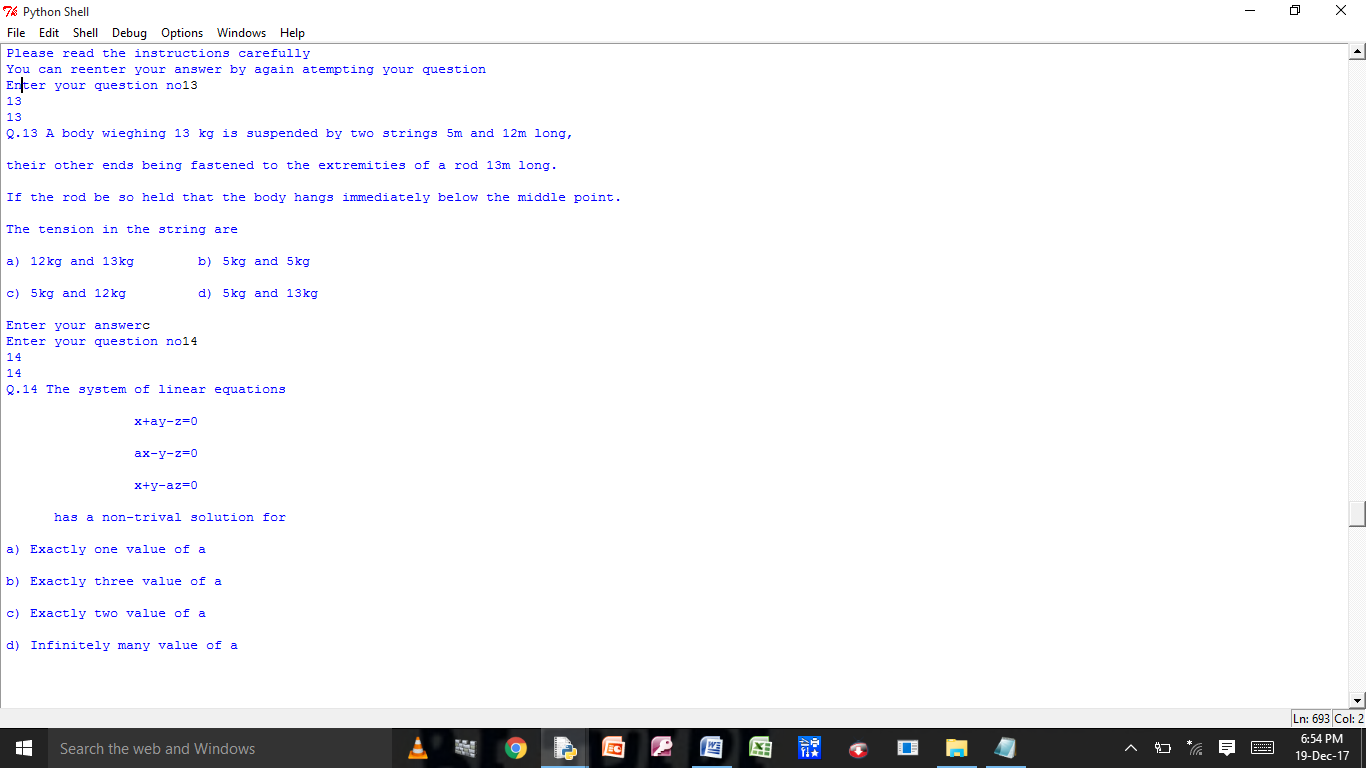


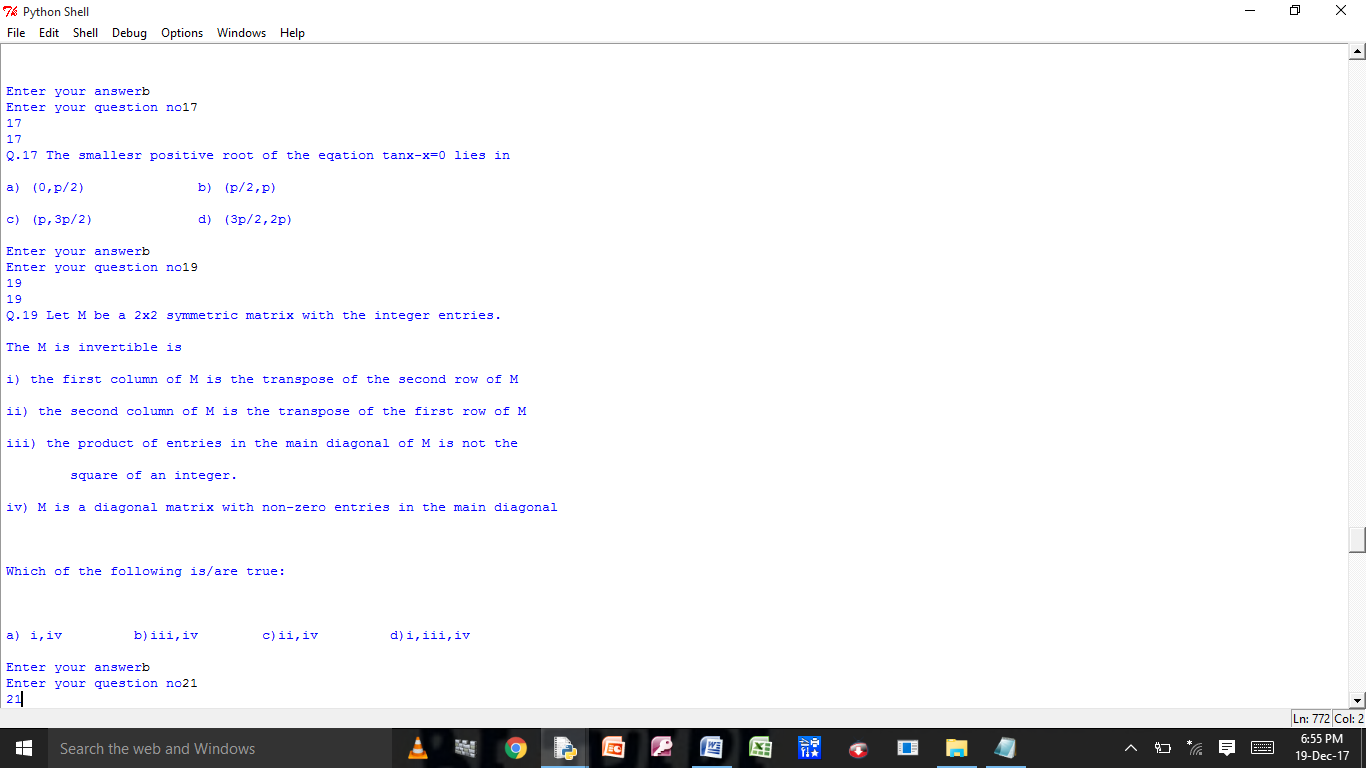


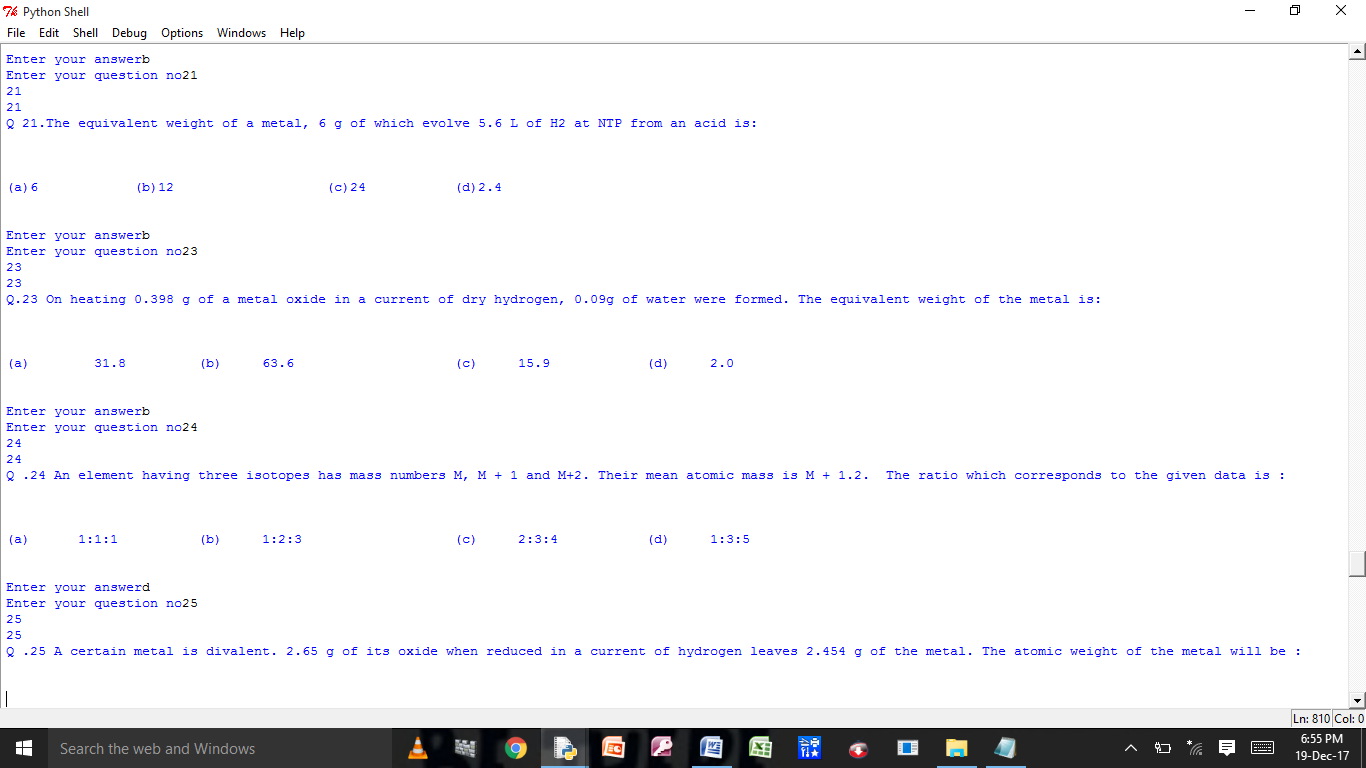


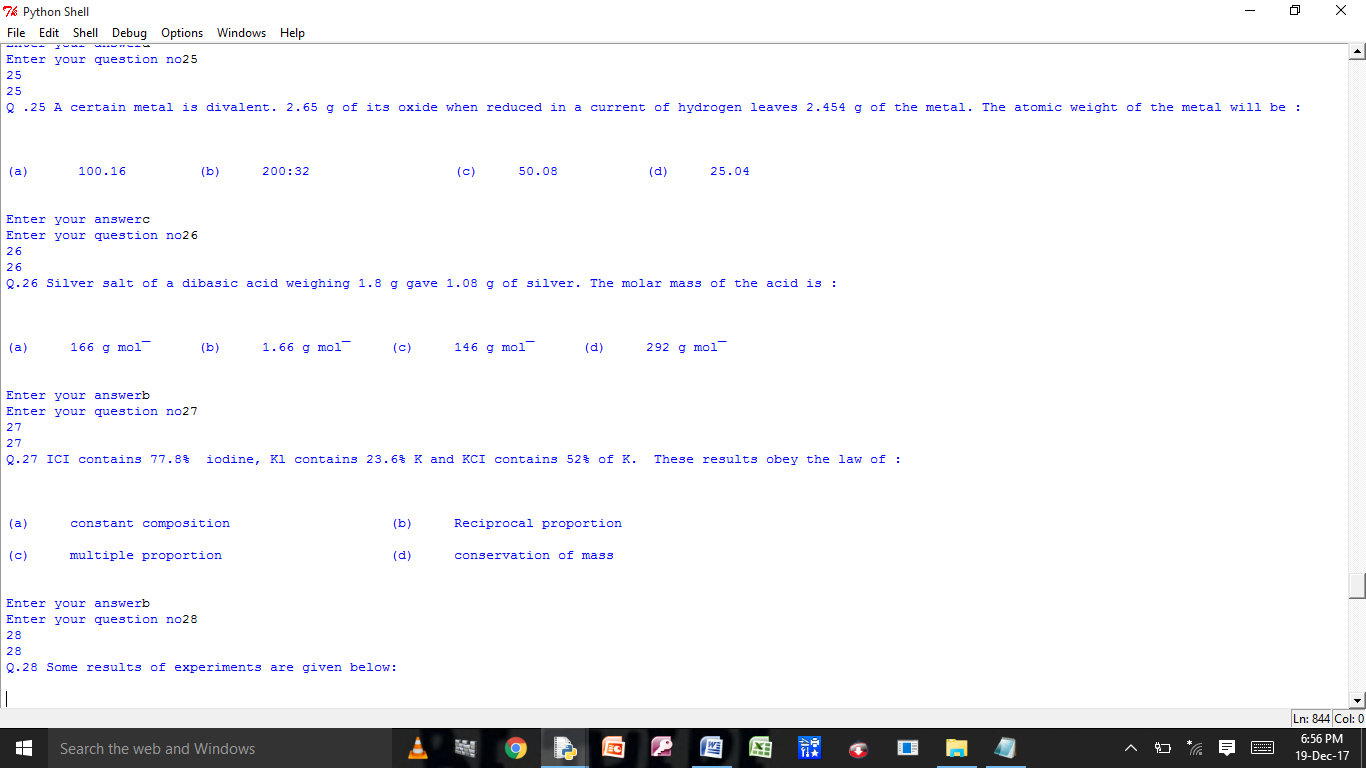


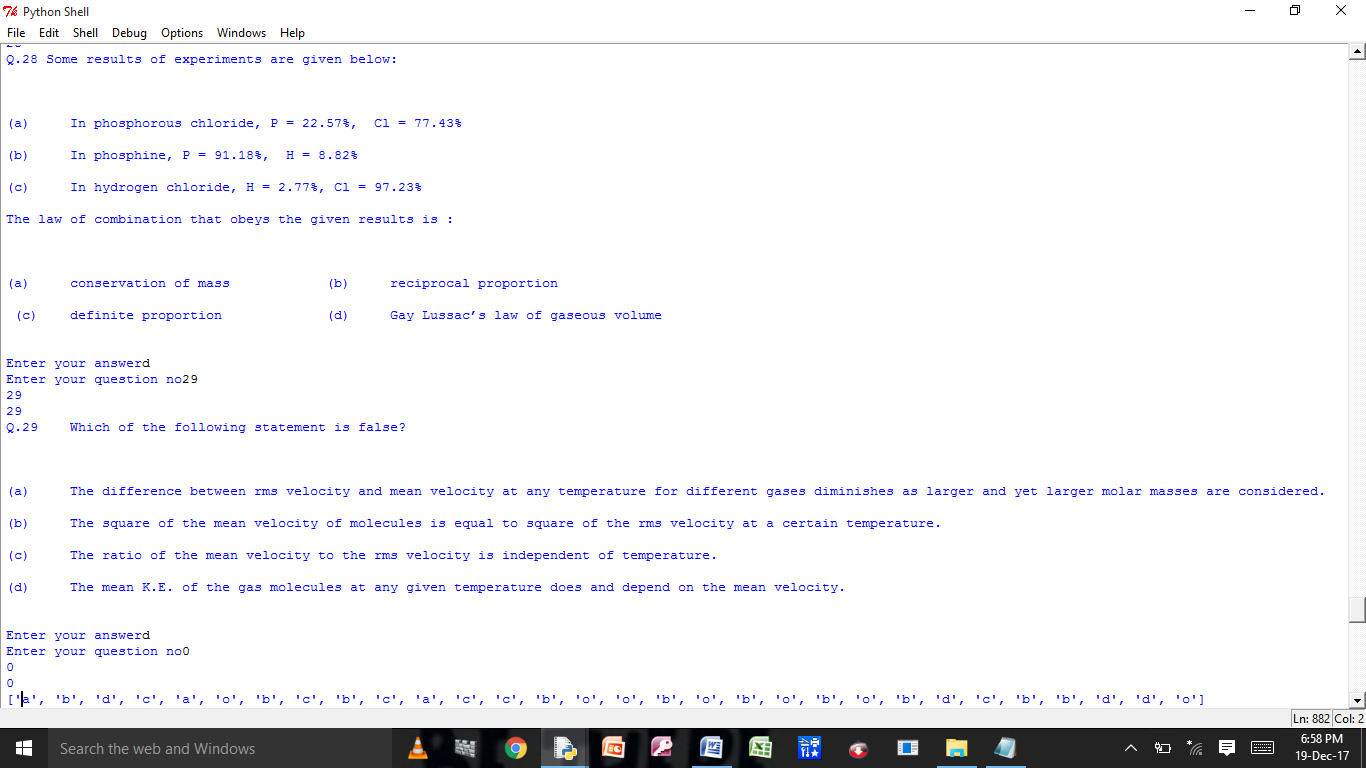


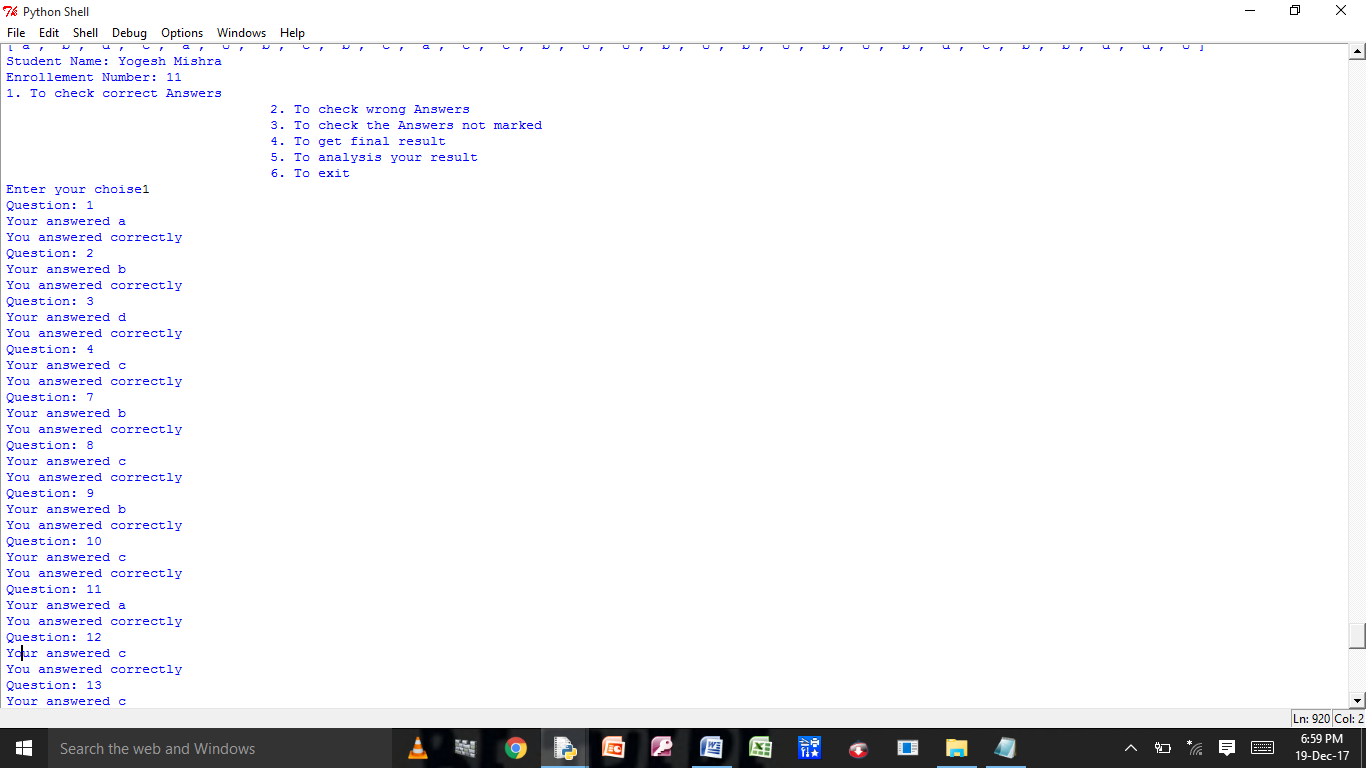


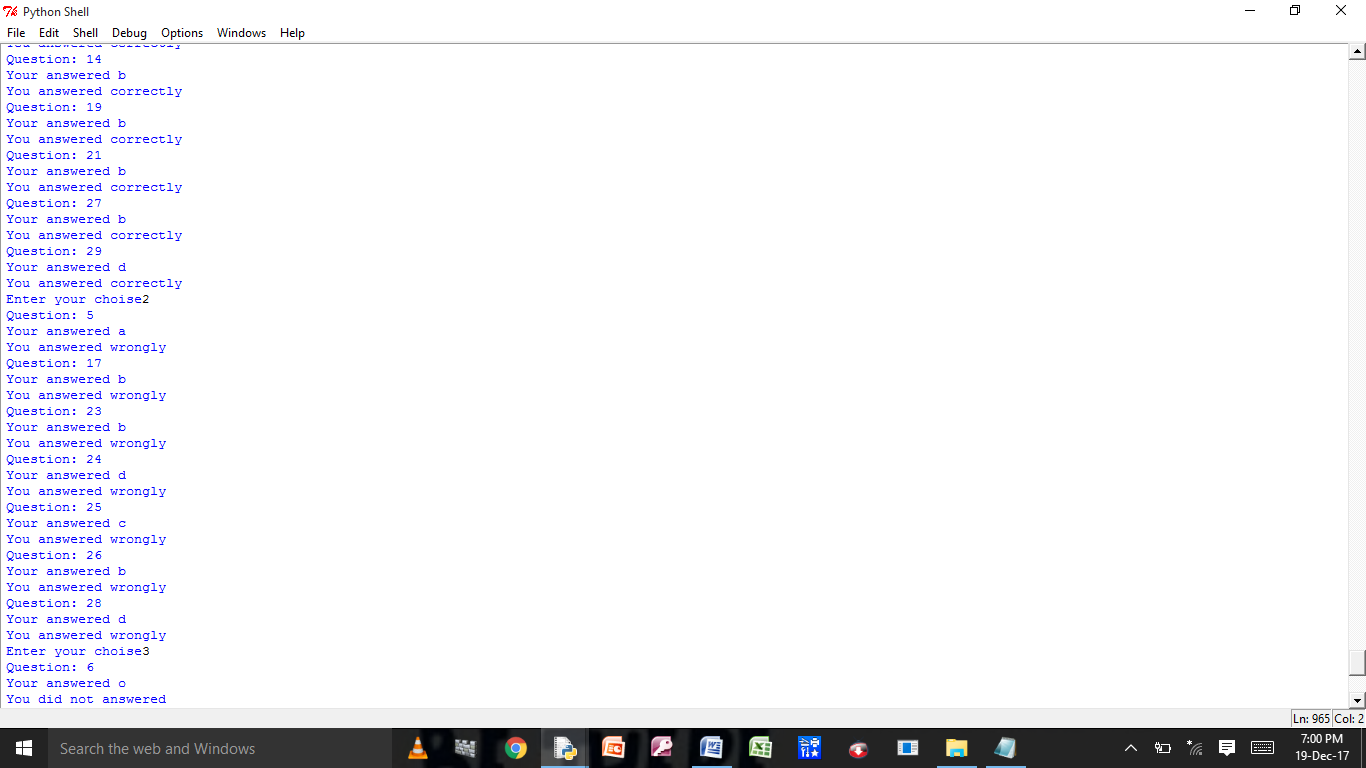


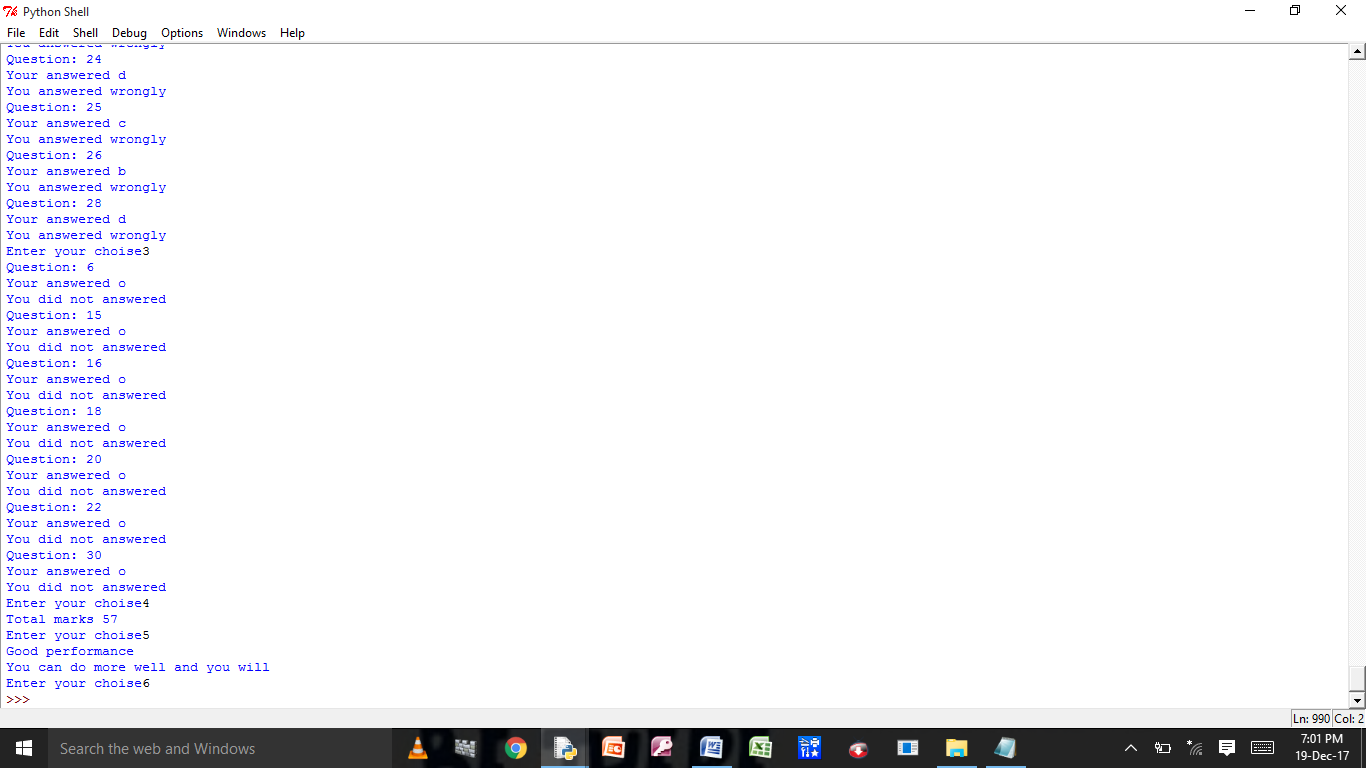












***Bibliography***

1. NCERT: Computer Science (Python) class 11th
2. NCERT : Computer Science(Python) class 12th
3. Computer Science with python (XI)
4. Computer Science with python (XII)

-----------------------------------------------------------------------------------------------