Python Programming

Department Elective - III

Subject Code : 2180711

Teaching Scheme

Teaching and Examination Scheme:

Teaching Scheme			Credits	Examination Marks						Total Marks
				Theory Marks			Practical Marks			
L	T	P	C	ESE (E)	PA (M)		ESE (V)		PA (I)	
					PA	ALA	ESE	OEP		
3	0	2	5	70	20	10	20	10	20	150

Syllabus

Sr. No.	Content	Total Hrs	% Weightage
1	Introduction to Python • The basic elements of python • Branching Programs • Control Structures • Strings and Input • Iteration	4	7%
2	Functions, Scoping and Abstraction • Functions and scoping • Specifications • Recursion • Global variables • Modules • Files • System Functions and Parameters	5	10%
3	Structured Types, Mutability and Higher-Order Functions • Strings, Tuples, Lists and Dictionaries • Lists and Mutability • Functions as Objects	4	8%
4	Testing, Debugging, Exceptions and Assertions • Types of testing – Black-box and Glass-box • Debugging • Handling Exceptions	4	7%

Syllabus

	• Assertions		
5	Classes and Object-Oriented Programming • Abstract Data Types and Classes • Inheritance • Encapsulation and Information Hiding	4	8%
6	Simple Algorithms and Data structures • Search Algorithms • Sorting Algorithms • Hash Tables	5	10%
7	Advanced Topics I • Regular Expressions – REs and Python • Plotting using PyLab • Networking and Multithreaded Programming – Sockets, Threads and Processes, Chat Application	10	20%
8	Advance Topics II • Security – Encryption and Decryption , Classical Cyphers • Graphics and GUI Programming – Drawing using Turtle, Tkinter and Python, Other GUIs	12	30%

Reference Books

1. John V Guttag. "Introduction to Computation and Programming Using Python", Prentice Hall of India
2. R. Nageswara Rao, "Core Python Programming", dreamtech
3. Wesley J. Chun. "Core Python Programming - Second Edition", Prentice Hall
4. Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, "Data Structures and Algorithms in Pyhon", Wiley
5. Kenneth A. Lambert, "Fundamentals of Python – First Programs", CENGAGE Publication
6. Luke Sneeringer, "Professional Python", Wrox
7. "Hacking Secret Ciphers with Python", Al Sweigart, URL- https://inventwithpython.com/hacking/chapters

Why Python?

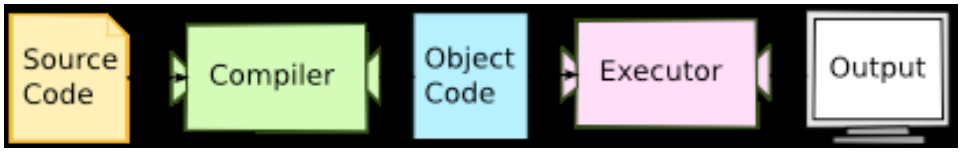
- Programmers want C style coding
- Java style object orientation
- Answer is Python
- Python is a programming that combines the features of C and Java

Introduction to Python

- Python is an interpreted programming language
- A program is a set of instructions telling the computer what to do.
- It has a strict syntax, and will only recognize very specific statements. If the interpreter does not recognize what you have typed, it will complain until you fix it.

Compiled vs. Interpreted

- Both Compilers and Interpreters translate source code to machine language



- Compiled
  - Must compile program on each target CPU architecture prior to execution.

- Interpreted
  - Code can be directly run on any platform that has an available interpreter

Python…

- Python is general purpose programming language
  - Almost any kind of program that does not need direct access to the computer's hardware
- Guido von Rossum in 1990
  - Python 2.0 in 2000
  - Python 3.0 in 2008
  - Now Python 3.5

Features of Python

- Simple
  - Like reading English sentences
- Easy to learn
  - Very few keywords
- Open Source
- High Level Language
- Dynamically typed
  - No declaration
  - Assignment binds a name to an object, object can be of any type
  - Once name is assigned it may be assigned to different type also

Features of Python…

- Platform independent
  - PVM
- Portable
  - When a program yields the same result on any computer in the world
- Procedure and object oriented

Python

- Interpreted
  - A program code is called source code
  - After writing a Python program, we should compile the source code using Python compiler
  - Python compiler translates the Python program into intermediate code called byte code
  - Byte code is executed by PVM.
  - Inside PVM, an interpreter converts the byte code instructions into machine code

Python…

- Extensible
  - The programs or pieces of code written in C or C++ can be integrated into Python and executed using PVM
  - Jython is useful to integrate Java code into Python programs and run on JVM
  - IronPython - >net

Python…

- Embeddable
  - We can insert Python programs into a C or C++ program
- Huge Library
- Scripting Language
  - It is a language that does not uses a compiler for executing the source code
  - It uses interpreter to translate source code into machine code on the fly

Python…

- Database connectivity
  - Oracle, Sybase, MySql
- Scalable
- Batteries Included
  - Small applications
  - packages

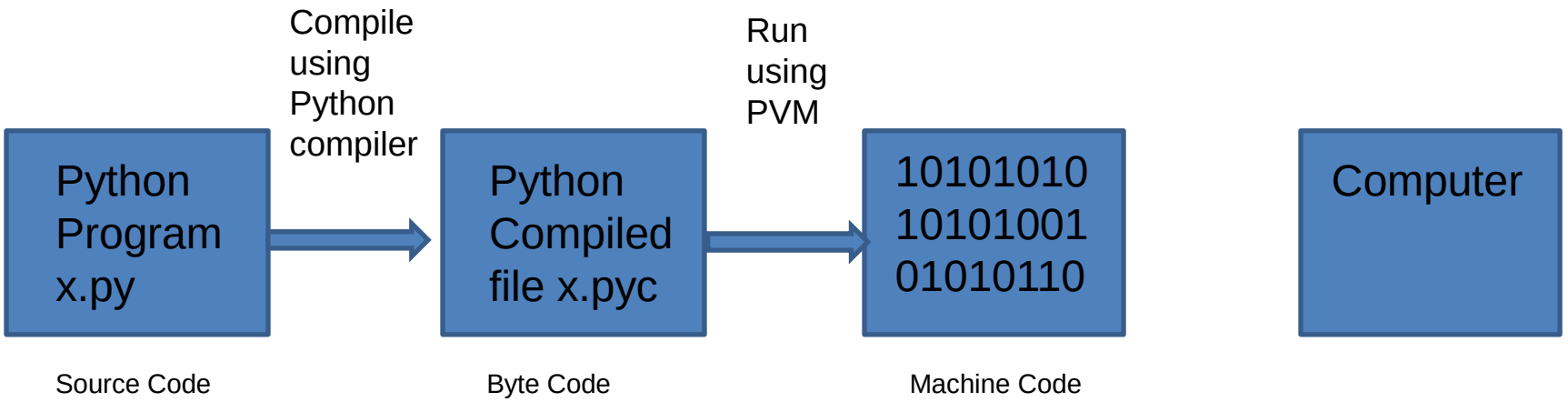
Batteries Included

- argparse – command line parsing library
- boto is Amazon web service library
- CherryPy is an object-oriented HTTP Framework
- cryptography
- Fiona – big data files
- jellyfish – approximate and phonetic matching of string
- mysql-connector-python – driver to connect to MySQL
- numpy – arrays of single and multidimensional

Batteries Included…

- pandas- powerful data structures for data analysis
- Pillow – imaging library
- pyquery – jquery-like library
- Scipy – scientific library – engineering
- Sphinx – python documentation generator
- Sympy – computer algebra system
- W3lib – web related functions
- Whoosh – fast and pure Python full text indexing, search and spell checking library

Execution of a Python Program



- C:\>python x.py
- Source code ->byte code -> machine code ->output

Flavors of Python

- CPython – implemented in C
- Jython – implemented in Java
- IronPython - .Net
- PyPy – Python implementation using Python
- RubyPython – bridge between the Ruby and Python interpreters
- StacklessPython – Small tasks which should run individually are called tasklets
- Pythonxy – after adding scientific and engineering related packages
- AnacondaPython – When Python is redeveloped for handling large-scale data processing, predictive analytics and scientific computing

Frozen Binaries

- When s/w is developed in Python there are two ways to deliver
- First
  - .pyc files to user
  - User will install PVM in pc
  - Run byte code instructions of the .pyc files
- Second
  - .pyc files, PVM along with necessary Python library
  - All .pyc files, related Python library and PVM will be converted into a single ..exe
- This is called frozen binaries

Python IDE's

- Typing programs directly into the shell is highly inconvenient.
- Most programmers prefer to use some sort of text editor that is part of an **integrated development environment** (**IDE**).
- Anaconda and Canopy are among the more popular of these IDE's

Python IDE's...

- Python IDE's provide:
- text editor with syntax highlighting, auto completion, and smart indentation
- shell with syntax highlighting, and
- an integrated debugger, which you should ignore for now.

Python IDE's…

The **file menu** includes commands to

- create a new editing window into which you can type a Python program,
- open a file containing an existing Python program, and
- save the contents of the current editing window into a file (with file extension .py).

https://www.enthought.com/products/canopy/

Python IDE's…

- The **edit menu** includes standard text-editing commands (e.g., copy, paste, and find) plus some commands specifically designed to make it easy to edit Python code

C / Python

C	Python
Procedure oriented. Does not have classes, objects, inheritance, polymorphism etc.	Object oriented. Contains all features.
Program execute faster.	Python programs are slower compared to C. PyPy can run faster.
Compulsory to declare data types, variables, arrays.	Type declaration is not required.
Pointers are there.	Pointers are not there.
Does not have exception handling.	Have exception handling and hence robust.
Do, do—while, for loops.	Have while and for loops.
Has switch statement.	Does not have switch.
Variable in the loop does not increment automatically.	Variable in the for loop increments automatically.
Memory management to be done.	Automatically done.

C / Python…

C	Python
Does not contain garbage collector.	Automatic garbage collector.
Array index should be positive integer.	Array index can be positive, negative integer.
Supports single and multi dimensional arrays.	Only single dimensional array. Third-party application like numpy.
Indentation not necessary.	It is necessary.
Semicolon is used.	New line indicates end of statements. Semicolon is used as an expression separator.
Supports in-line assignments.	Does not support in-line assignments.

The basic elements of Python

- Python program, sometimes called a script
  - Sequence of definitions and commands
- Python interpreter is called shell
- A command often called a statement
- The symbol >>> is shell prompt

Objects, Expressions and Numerical Types

- Objects are core things that Python programs manipulate.
- Every object has a type
  - Scalar – objects are indivisible
  - Non-scalar – objects have internal structure – string

Scalar objects

- int – represent integers
- float – real numbers
- bool – boolean values True and False
- None – single value

Example

```
>>> 3 + 2
5
>>> 3.0 + 2.0
5.0
>>> 3 != 2
True
```

Example…

```
>>> type(3)
<type 'int'>
>>> type(3.0)
<type 'float'>
```

Operators on types int and float

- **i+j** is the sum of i and j. If i and j are both of type int, the result is an int. If either of them is a float, the result is a float.
- **i–j** is i minus j. If i and j are both of type int, the result is an int. If either of them is a float, the result is a float.
- **i*j** is the product of i and j. If i and j are both of type int, the result is an int. If either of them is a float, the result is a float.

Operators on types int and float...

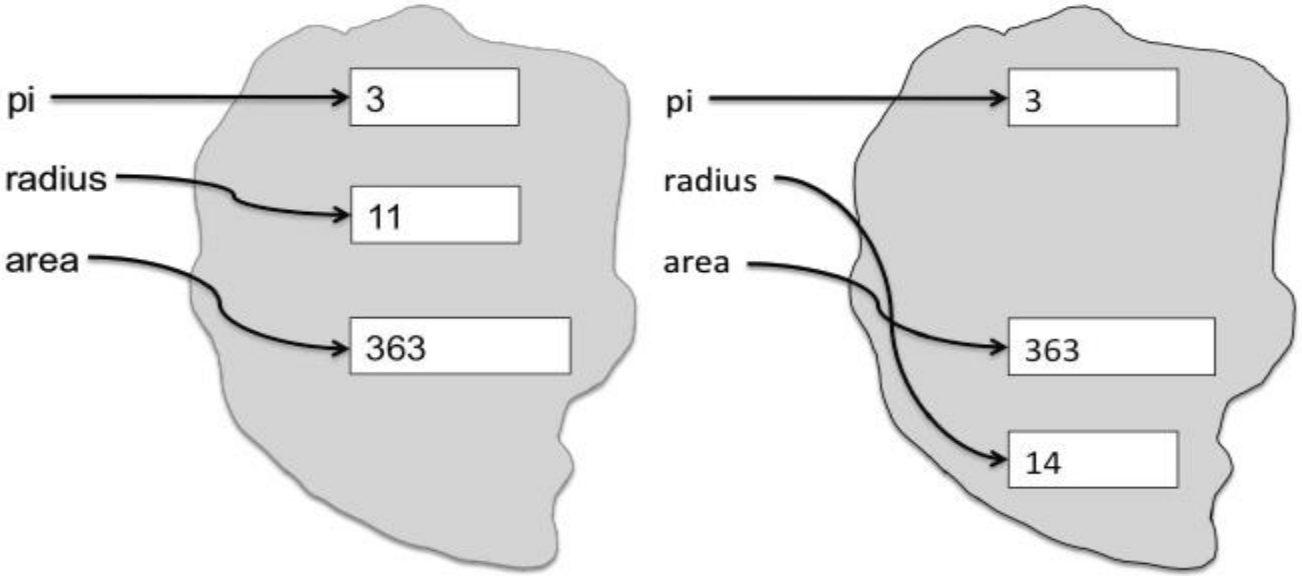
- **i//j** is integer division. For example, the value of 6//2 is the int 3 and the value of 6//4 is the int 1. The value is 1 because integer division returns the quotient and ignores the remainder.
- **i/j** is i divided by j. In Python 2.7, when i and j are both of type int, the result is also an int, otherwise the result is a float. In this book, we will never use / to divide one int by another. We will use // to do that. (In Python 3, the / operator, thank goodness, always returns a float. For example, in Python 3 the value of 6/4 is 1.5.)

Operators on types int and float...

- **i%j** is the remainder when the int i is divided by the int j. It is typically pronounced “i mod j,” which is short for “i modulo j.”
- **i**j** is i raised to the power j. If i and j are both of type int, the result is an int. If either of them is a float, the result is a float.
- The comparison operators are == (equal), != (not equal), **>** (greater), **>=** (at least), **<**, (less) and **<=** (at most).

Variables and Assignment

- pi = 3
- radius = 11
- area = pi * (radius**2)
- radius = 14



Variable…

- In Python, a variable name is just a name

Consider the two code fragments

- a = 3.14159 pi = 3.14159
- b = 11.2 diameter = 11.2
- c = a*(b**2) area = pi*(diameter**2)
- In Python, variable names can contain uppercase and lowercase letters, digits (but they cannot start with a digit), and the special character _.

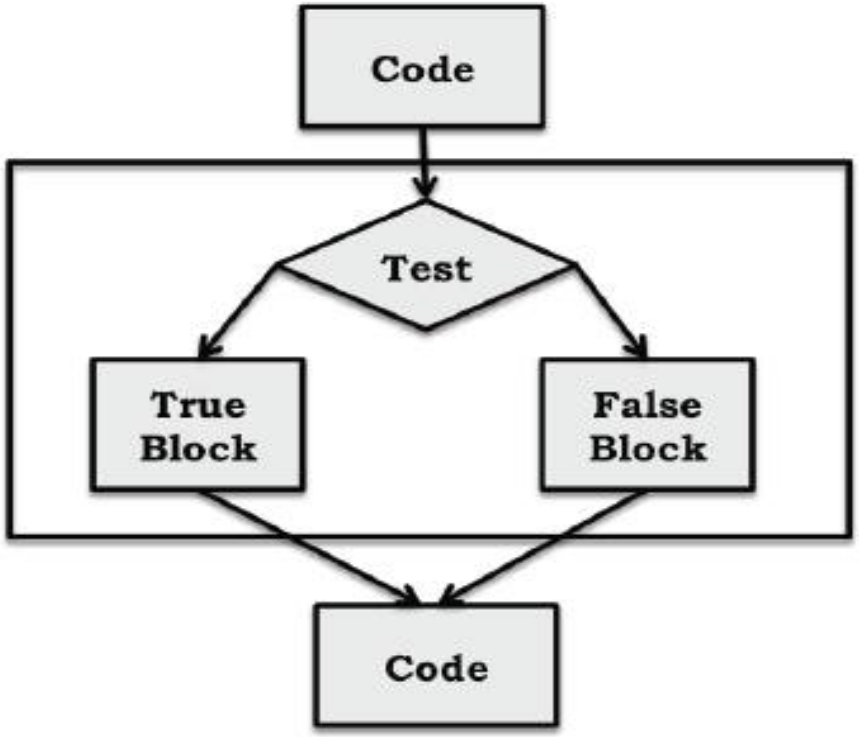
Reserved words

- Different versions of Python have slightly different lists of reserved words.
- The reserved words in Python 2.7 are and, as, assert, break, class, continue, def, del, elif, else, except, exec, finally, for, from, global, if, import, in, is, lambda, not, or, pass, print, raise, return, try, with, while, and yield.

Comments

- Another good way to enhance the readability of code is to add **comments**.
- Text following the symbol # is not interpreted by Python.
- #subtract area of square s from area of circle c
- areaC = pi*radius**2
- areaS = side*side
- difference = areaC-areaS
- Python allows multiple assignment.
- The statement x, y = 2, 3

Branching Programs



Branching Programs…

```
if x%2 == 0:
        print 'Even'
else:
        print 'Odd'
print 'Done with conditional'
```

- The expression x%2 == 0 evaluates to True when the remainder of x divided by 2 is 0, and False otherwise.
- Remember that == is used for comparison, since = is reserved for assignment.

Branching Programs…

- **Indentation** is semantically meaningful in Python.
- if the last statement in the above code were indented
  - it would be part of the block of code associated with the else,
  - rather than with the block of code following the conditional statement.

Branching Programs…

```
if x%2 == 0:
        if x%3 == 0:
                print 'Divisible by 2 and 3'
        else:
                print 'Divisible by 2 and not by 3'
elif x%3 == 0:
        print 'Divisible by 3 and not by 2'
```

The elif in the above code stands for “else if.”

Branching Programs…

It is often convenient to use compound Boolean expressions in the test of a conditional, for example,

```
if x < y and x < z:
        print 'x is least'
elif y < z:
        print 'y is least'
else:
        print 'z is least'
```

- Conditionals allow us to write programs that are more interesting than straight line programs, but the class of branching programs is still quite limited.

Strings and Input

- Objects of type str are used to represent strings of characters. Literals of type str can be written using either single or double quotes, e.g., 'abc' or "abc".
- The literal '123' denotes a string of characters, not the number one hundred twenty-three.
- remember that the >>> is a prompt, not something that you type:

```
>>> 'a‘
>>> 3*'a'
>>> 'a'+'a'
```

```
>>> 3*4
>>> 3+4
```

The operator + is said to be **overloaded**:

It has different meanings depending upon the types of the objects to which it is applied.

Strings and Input…

- The **length** of a string can be found using the len function. For example, the value of len('abc') is 3.
- **Indexing** can be used to extract individual characters from a string.
- In Python, all indexing is zero-based.
- For example, typing 'abc'[0] into the interpreter will cause it to display the string 'a'. Typing 'abc'[3] will produce the error message IndexError: string index out of range.
- Since Python uses 0 to indicate the first element of a string, the last element of a string of length 3 is accessed using the index 2.
- Negative numbers are used to index from the end of a string. For example, the value of 'abc'[-1] is 'c'.

Strings and Input…

- **Slicing** is used to extract substrings of arbitrary length. If s is a string, the expression s[start:end] denotes the substring of s that starts at index start and ends at index end-1.
- For example, 'abc'[1:3] = 'bc'. Why does it end at index end-1 rather than end?
- So that expressions such as 'abc'[0:len('abc')] have the value one might expect.
- If the value before the colon is omitted, it defaults to 0. If the value after the colon is omitted, it defaults to the length of the string.
- Consequently, the expression 'abc'[:] is semantically equivalent to the more verbose 'abc'[0:len('abc')].

Input

```
>>> name = input('Enter your name: ')
Enter your name: George Washington
>>> print('Are you really', name, '?‘)
Are you really George Washington ?
>>> print('Are you really ' + name + '?‘)
Are you really George Washington?
```

Input…

```
>>> n = input('Enter an int: ')
Enter an int: 3
>>> print type(n)
<type 'str'>
```

- Notice that the variable n is bound to the str '3' not the int 3.
- So, for example, the value of the expression n*4 is '3333' rather than 12.
- string is a valid literal of some type, a type conversion can be
- **Type conversions** (also called **type casts**) are used often in Python code.
- So, for example, the value of int('3')*4 is 12. When a float is converted to an int, the number is truncated (not rounded), e.g., the value of int(3.9) is the int 3.

A digression about character encoding

- For many years most programming languages used a standard called ASCII
- In recent years, shift to Unicode
- Unicode standard is a character coding system designed to support the digital processing and display of the written texts of all languages.
- It contains more than 120,000 different characters – covering 129 modern and historic scripts

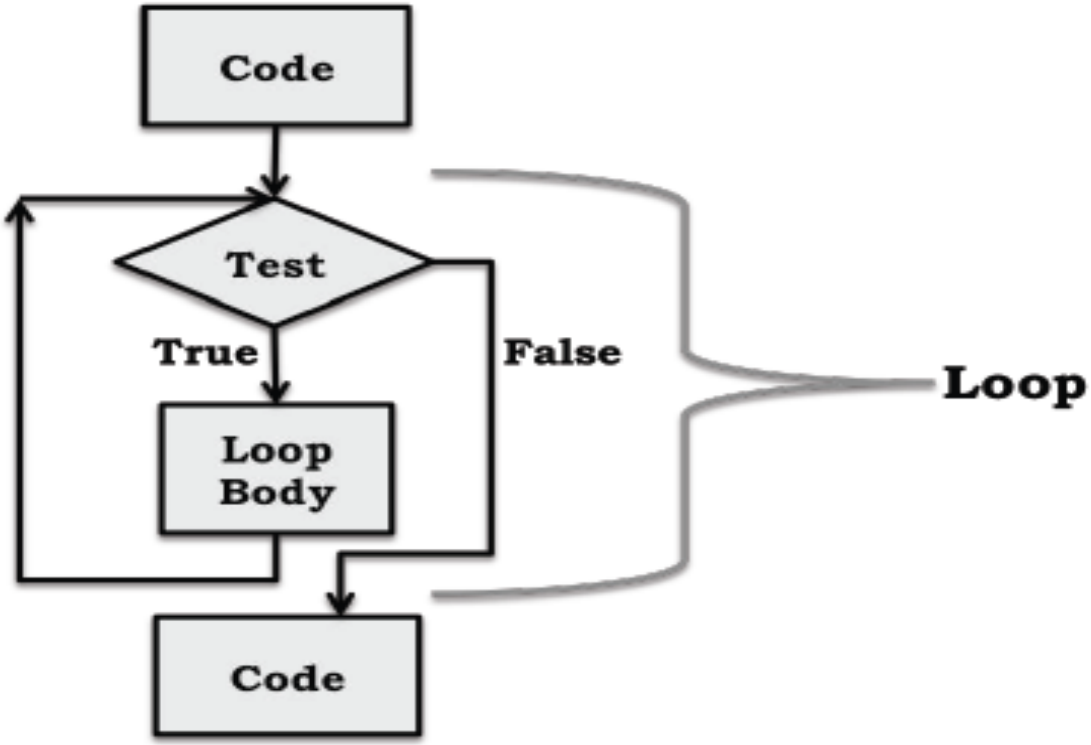
A digression about character encoding…

#_*_ coding: encoding name _*_

#_*_ coding: utf-8 _*_

- UTF-8 is the most frequently used character encoding for WWW pages.

Iteration



Iteration…

```
numXs = int(input('How many times should print x?'))
toPrint = ' '
if numXs  == 1:
        toPrint = 'X'
elif numXs == 2:
        toPrint = 'XX'
elif numXs == 3:
        toPrint = 'XXX'
#...
Print(toPrint)
```

Iteration…

```
numXs = int(input('How many times should print x?'))
toPrint = ' '
concatenate X to toPrint numXs times
print(toPrint)
```

Iteration…

```
# Square an integer, the hard way
x = 3
ans = 0
itersLeft = x
while (itersLeft != 0):
        ans = ans + x
        itersLeft = itersLeft - 1
Print(str(x) + '*' + str(x) + ' = ' + str(ans))
```

Iteration…

test #	x	ans	itersLeft
1	3	0	3
2	3	3	2
3	3	6	1
4	3	9	0

Program to find grades of student

```
number = input("Enter the numeric grade: ")
if int(number) > 89:
     letter = 'A'
elif int(number) > 79:
     letter = 'B'
elif int(number) > 69:
     letter = 'C'
else:
     letter = 'F'
print ("The letter grade is", letter)
```

While Loop

```
while True:
      number = input("Enter the numeric grade: ")
      if int(number) >= 0 and int(number) <= 100:
            break
      else:
            print ("Error: grade must be between 100
            and 0")
print (number)
```

Summation of given numbers

```
sum = 0.0
data = input("Enter a number: ")
while data != "":
      number = float(data)
      sum += number
      data = input("Enter the next number: ")
print ("The sum is", sum)
```

Encryption

```
plainText = input("Enter a one-word, lowercase message:")
distance = input("Enter the distance value: ")
code = ""
for ch in plainText:
      ordValue = ord(ch)
      cipherValue = ordValue + int(distance)
      if cipherValue > ord('z'):
            cipherValue = ord('a') + int(distance) -
            (ord('z') - ordValue + 1)
      code += chr(cipherValue)
print (code)
```

Binary to Decimal

```
bstring = input("Enter a string of bits: ")
decimal = 0
exponent = len(bstring) - 1
for digit in bstring:
      decimal = decimal + int(digit) * 2 ** exponent
      exponent = exponent - 1
print ("The integer value is", decimal)
```

Decimal to Binary

```
decimal = input("Enter a decimal integer: ")
if decimal == 0:
      print ("0")
else:
      print ("Quotient Remainder Binary")
      bstring = ""
      while int(decimal) > 0:
            remainder = int(decimal) % 2
            decimal = int(decimal) / 2
            bstring = str(remainder) + bstring
            #print ("%5d%8d%12s" % (decimal,
            remainder, bstring))
      print ("The binary representation is", bstring)
```

Thank you!!!