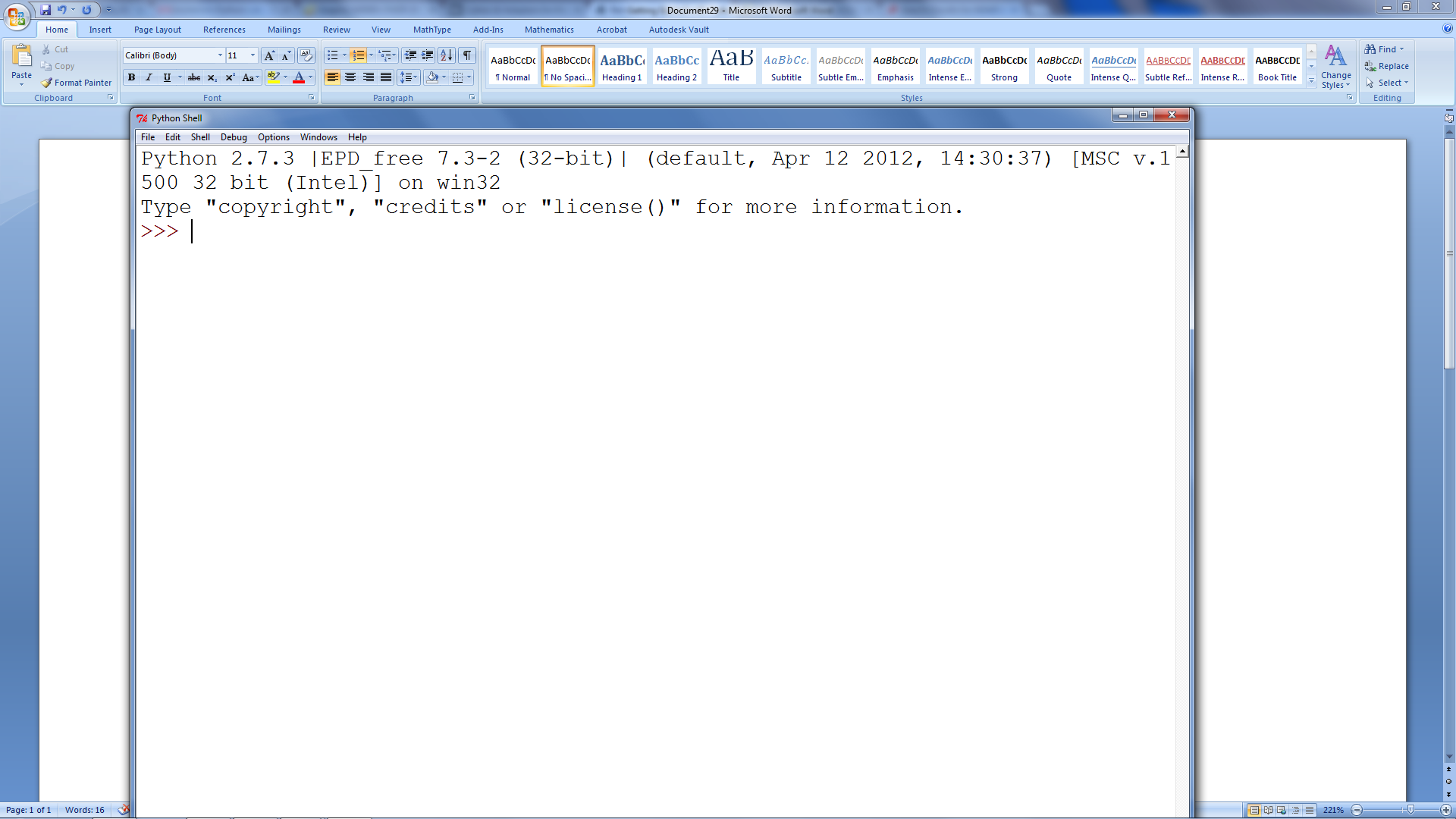
**Python Basics**

**Using IDLE and the Python Shell**

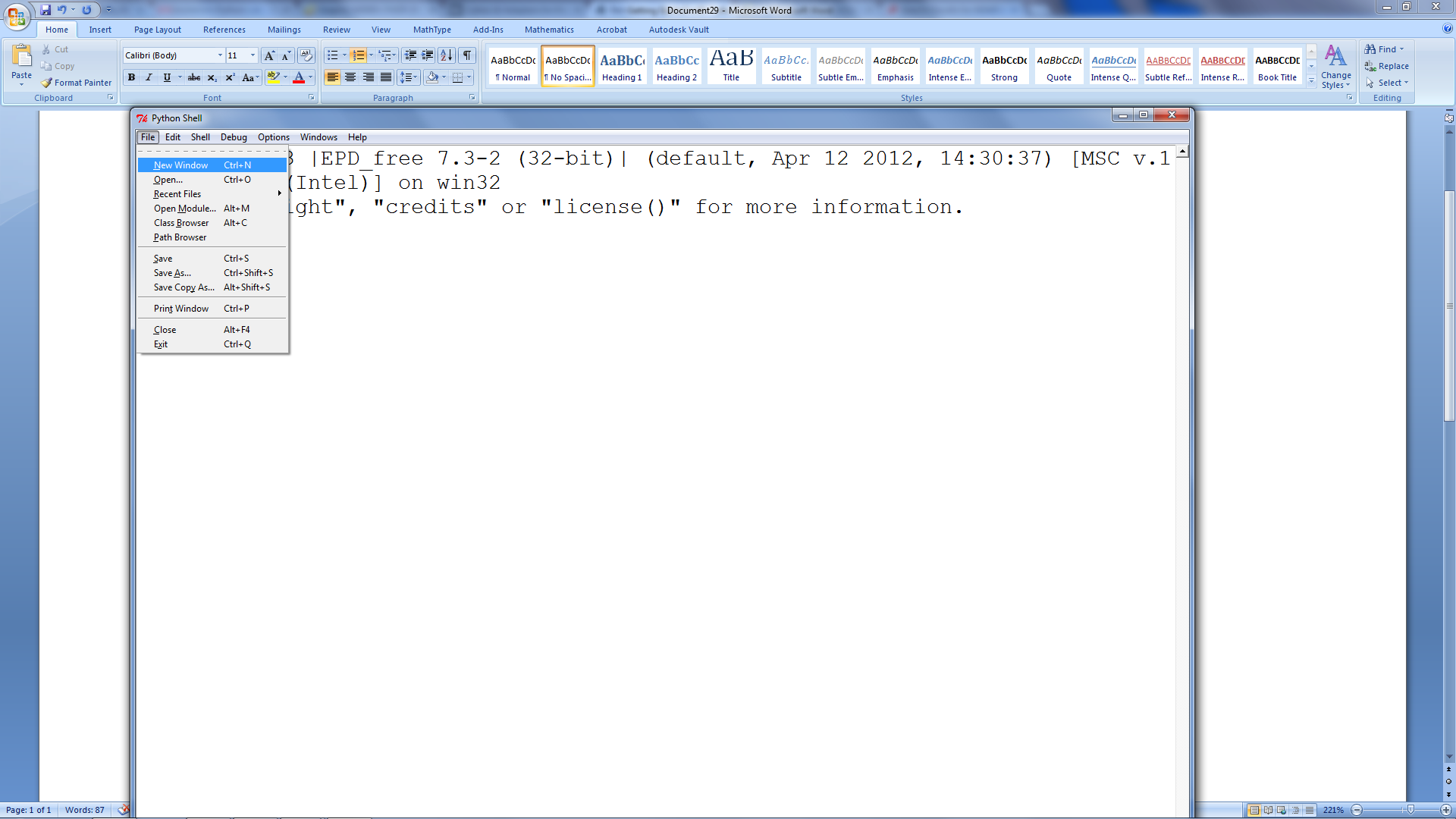
1. Start IDLE by clicking on the IDLE icon on the desktop to open the Python shell.



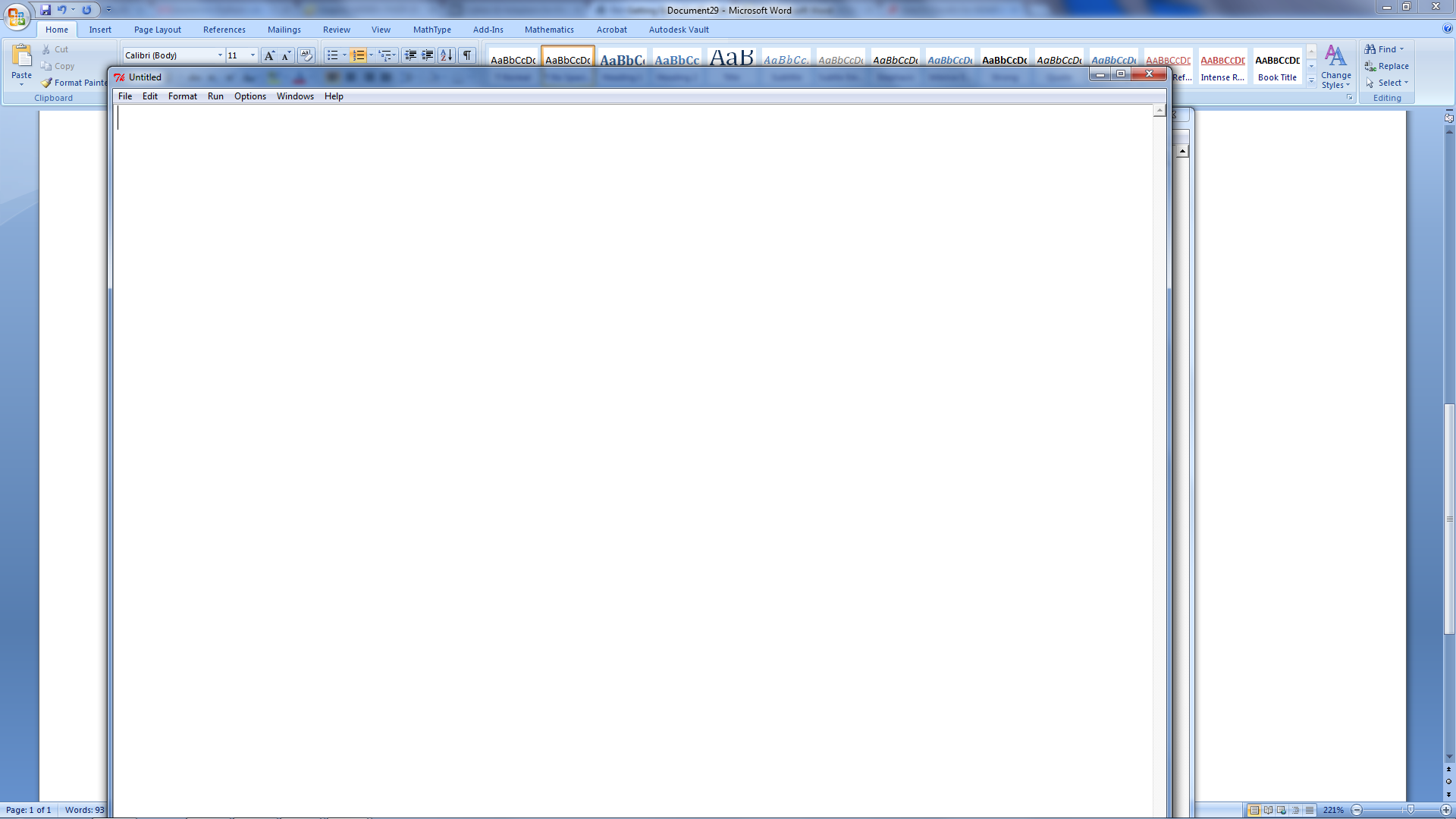
1. This is an interactive environment which we may use to issue Python commands.
2. At the >>> prompt, issue the following commands:
   1. print 'Hello World'
   2. 87 \* 34
   3. 87 \*\* 34
   4. 7 / 3
   5. 7 % 3
   6. 7 / 3.0
   7. import math
   8. math.sqrt(7)
   9. 7 \*\* 0.5

**Using IDLEs Editor**

1. Choose File|New Window to open an editor window



1. This will open up a window where we can write and run our Python programs.



1. Type the following code into the window.

print 'Hello World'

print 87 \* 34

print 87 \*\* 34

print 7 / 3

print 7 % 3

print 7 / 3.0

import math

print math.sqrt(7)

print 7 \*\* 0.5

1. You should notice that this is the same output as when you typed commands into the Python shell. The only difference was that you needed to tell Python to print.

Example Program 1

This program will tell you how many days until your birthday. It will demonstrate several key Python concepts:

* Module import
* Using raw\_input function to get user input (data type string)
* Splitting a string (creates a list)
* Using the Date class to retrieve the current date and time
* Using the Date class to construct a Date object
* Using a built-in Python function (abs)
* Using a decision making structure to handle the situation that you have already had your birthday this hear so you will calculate the days to your birthday next year.

from datetime import date

bd = raw\_input("Enter your birthday in the form mm/dd/yyyy: ")

bday = bd.split('/')

today = date.today()

my\_birthday = date(today.year, int(bday[0]), int(bday[1]))

if my\_birthday < today:

my\_birthday = my\_birthday.replace(year=today.year + 1)

time\_to\_birthday = abs(my\_birthday - today)

print 'It is', time\_to\_birthday.days, 'days until your birthday'

Your turn

1. Rewrite the program to calculate how many days old you are.
2. Rewrite the program to calculate the number of days of Summer vacation you have left.

Bonus

Here is some code that will tell you how many seconds until your birthday:

from datetime import \*

bd = raw\_input("Enter your birthday in the form mm/dd/yyyy: ")

bday = bd.split('/')

today = datetime.today()

my\_birthday = datetime(today.year, int(bday[0]), int(bday[1]))

if my\_birthday < today:

my\_birthday = my\_birthday.replace(year=today.year + 1)

time\_to\_birthday = abs(my\_birthday - today)

print 'It is', time\_to\_birthday.seconds, 'seconds until your birthday'

Note: The only difference is that we used a Datetime object instead of a Date object.

**Working with Numbers**

1. Numbers are manipulated using arithmetic operators (\*, /, +, -, \*\*, and %).
2. Google was named after the Googol which is a number 1 followed by one hundred 0s. A googolplex is 10 raised to the googol power. Use the \*\* operator to see if Python can calculate these.

In the PBS science program *Cosmos: A Personal Voyage*, Episode 9: "The Lives of the Stars", astronomer and television personality Carl Sagan estimated that writing a googolplex in standard form (i.e., "10,000,000,000...") would be physically impossible, since doing so would require more space than is available in the known universe.

A typical book can be printed with 106 zeros (around 400 pages with 50 lines per page and 50 zeros per line). Therefore it requires 1094 such books to print all zeros of googolplex.

Writing the number takes too long: if a person can write two digits per second, then writing a googolplex would take around about 1.51×1092 years, which is about 1.1×1082 times the age of the universe.

**How Fast is your Raspberry Pi**

We are going to test how fast your Raspberry Pi is. To do this we will ask Python to calculate the square root of a number one million times and see how long it takes. On an I7 Windows 7 machine with 12 GB of RAM it takes around 0.35 seconds. Here is some code that will let you measure it.

import time, math

start = time.time()

for i in range(1000000):

number = math.sqrt(i)

end = time.time()

duration = end - start

print "You calculated one million square roots in", duration, "seconds"

**Pascal's Triangle**

Here is a little program that will print Pascal's Triangle.

def pascal(n):

row = [1]

k = [0]

for x in range(n):

for i in range(n-x,-1,-1):

print ' ',

print row

row=[l+r for l,r in zip(row+k,k+row)]

pascal(10)

Here is a second version which uses some of Python's formatting function

def MakeTriangle(numberOfRows):

triangle=[[1]] #base case (i.e., 0th row)

for x in range(numberOfRows - 1):

zipperd = zip([0] + triangle[-1], triangle[-1] + [0])

newRow = map(sum, zipperd)

triangle.append(list(newRow))

return triangle

for row in MakeTriangle(10):

print('{0:^50}'.format(row))

**Date formatting**

The following program uses a dictionary lookup to convert the month number to the month name.

months = {1:'January', 2:'February', 3:'March', 4:'April',

5:'May', 6:'June', 7:'July', 8:'August', 9:'September',

10:'October', 11:'November', 12:'December'}

d = raw\_input("Enter the date in mm/dd/yyyy format: ")

date = d.split('/')

month = int(date[0])

day = date[1]

year = date[2]

print months[month] + ' ' + day + ', ' + year

**Don't Worry Be Happy**

The following two line program demonstrates the use of Unicode to print out something happy. There are 65, 536 symbols available to you in Unicode (216)

for i in range(1000):

print u'\u263A',

**Finding Pi**

Now that you have a Raspberry Pi, let's use it to find Pi using something called Monte Carlo simulation

import random

for size in [10, 100, 1000, 10000, 100000, 1000000]:

count = 0

for i in range(1, size + 1):

x = random.random()

y = random.random()

if x\*\*2 + y\*\*2 < 1:

count += 1

pi = 4.0 \* count/size

print "With", size, "trials Pi =", pi

**Igpay Atinlay Ranslatortay**

vowels = ['a', 'e', 'i', 'o', 'u']

def pigLatin(word):

newWord = word.lower()

if newWord[0] not in vowels:

pLatin = newWord[1:] + newWord[0] + "ay"

else:

pLatin = newWord + "hay"

return pLatin

phrase = raw\_input("Enter a phrase: ")

words = phrase.split()

for word in words:

print pigLatin(word),

**Prime Factorization**

def isPrime(number):

for i in range(2, number):

if number % i == 0:

return False

return True

def factor(number):

factors = []

if isPrime(number): #need to handle the case of a number is prime

factors.append(number)

return factors

for i in range(2, number):

while number % i == 0:

factors.append(i)

number /= i

return factors

number = input("Enter a number you want factored: ")

print "The prime factors are: " + str(factor(number))