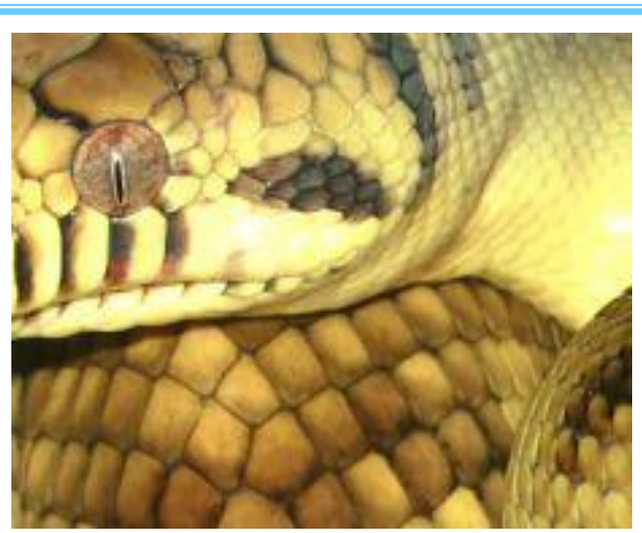
Project 1
Python Programming



Sources

- Exercises are partly based on MIT course MIT6_189 homework and Project Euler web site (http://projecteuler.net).

Problem 1: High Math

- Write a function `integrate(f,a,b,n=1000)` which accepts
  - A real function `f`
  - An interval points `a` and `b`
  - An optional division number `n` (defaults to 1000)
- It should compute the definite integral of f(x) over the interval `(a,b)` as a Rieman sum approximation.
- Try to test it for the functions:
  - `f1(x)=x**3`
  - `f2(x) = x**3*sin(x).`

Problem 2: DataBase processing

- Run this list comprehension in your prompt:

```
List1 = [x**2 for x in range(5,12)]
List2 = [x+y for x in [10,20,30,40] for y in [1,2,3,4]]
```

- Figure out what is going on here, and write a nested for loop that gives you the same result.
- Make sure what is going on makes sense to you!

Problem 3: Dictionary, quick ref

```
•  d = dict() creates an empty dictionary

* d = dict(key1=value1, key2=value2, ...)    - Create a new dictionary with initialized keys
  example:
      d = dict(name='Avi Cohen', age=32, id=5802231, address='Hayarden 43, Gedera')
      print "Avi's age is:", d['age']
      print "Avi's address is:", d['address']
      print "Avi has moved to a new town:"
      d['address'] = 'Hayarkon 25, Haifa‘

* The same thing can be achieved by:
      d = {key1: value1, key2: value2, ...}
  example:
      d = { 'name': 'Avi Cohen', 'age': 32, 'id': 5802231, 'address': 'Hayarden 43, Gedera' }
      or:
      pairs = [ ('name', 'Avi Cohen'), ('age', 32), ('id', 5802231),
                                                    ('address', 'Hayarden 43, Gedera') ]
      d = dict(pairs)

* d[key] - returns the value of the key. (What if there's no such key?)

* d[key] = newvalue - maps newvalue to key.
  Overwrites any previous value.
  Remember 'newvalue' can be any valid Python data structure

* del d[key] - deletes the mapping with that key from d.
* len(d) - returns the number of keys in d.
* x in d, x not in d - checks whether the key x is in the dictionary d.
* d.keys() - returns a list of all the keys in the dictionary.
* d.values() - returns a list of all the values in the dictionary.
```

Problem 3: Dictionary, quick ref

- Given two lists:

```
names = ['Alice', 'Bob', 'Cathy', 'Dan', 'Ed', 'Frank', 'Gary', 'Helen', 'Irene', 'Jack', 'Kelly', 'Larry']
ages  = [20, 21, 18, 18, 19, 20, 20, 19, 19, 19, 22, 19]
```

- These lists match up, so Alice's age is 20, Bob's age is 21, and so on.
- Write a function **`combine_lists`** that combines these lists into a dictionary
- Hint 1: what should the keys of this dictionary?
- Hint 2: what should be the values of this dictionary?
- Write a function **`people(age)`** that takes in an age and returns the names of all the people who are that age

Problem 3: Test Program (QA)

- Test your program's functions by running the following Python code:

```
print 'Dan' in people(18) and 'Cathy' in people(18)
print 'Ed' in people(19) and 'Helen' in people(19) and 'Irene' in people(19) \
           and 'Jack' in people(19) and 'Larry' in people(19)
print 'Alice' in people(20) and 'Frank' in people(20) and 'Gary' in people(20)
print people(21) == ['Bob']
print people(21) == ['Bob', 'Dan', 'Kelly']
print people(23) == []
```

Problem 4: Word Counting

- Download the text file: https://samyzaf.com/braude/PYTHON/projects/oliver_twist.txt
- (a) Write a function **`word_count(file)`** that prints the number of lines, words, and characters in a file. Test it on the above file:

```
>>> word_count("oliver_twist.txt")
file: oliver_twist.txt
19191 lines  160999 words  916980 characters
```

- (b) Write a function **`word_frequency(file)`** which counts how many times each word appears in that book.
- To make it simple: a word should consist only of English letters (no punctuation marks, hyphens, or quotes).
  - Hint: you should build a dictionary
  - Hint: Use Python string.punctuation to remove punctuation characters from words.

Problem 4: Word Counting

- Test your program by running it on oliver_twist.txt book (you should get 12733 words!)
- Try to sort the words by frequency (from most frequent to least frequent).

```
>>> word_frequency("oliver_twist.txt")
     730  Oliver
     303  gentleman
     288  Fagin
      53  Twist
      36  Crackit
       7  keyhole
       4  funny
     ......
(this is of course only a small part: there are 12733 words in this book!)
```

- (c) What is the most frequent 3 letters word in this book? How many times it appears in this book?
- (d) How many words occur more than 1000 times? (don't count your output, write a program to find this!)