Notes for Friday, Sept. 18 Ethan Macaulay

LERNING CENTRE (for Computer Science):
Room 233 of Computer Science building (2nd floor, accross form the elevators)
(Outside of Prof. Lushman's office
www.cs.dal.ca ----> services ------> Learning Centre
-Ask the TAs there for help when you get stuck

Office hours: Wed. 11-12

Midterm: Oct. 21, closed book

Assignment 1 is up! @ http://torch.cs.dal.ca/~bmlushman/csci1105/assns/a1.txt
Due Friday, Sept. 25

Recall from previous lecture: Volume of Pyramid function.

```
def volume_of_pyramid (b,h):
        return b**2*h/3
```

BUT

```
>>> volume_of_pyramid(1,1)
0 [correct answer is 0.333333...]
```

AND

```
>>> volume_of_pyramid(2,2)
2 [correct answer is 2.666666...]
```

Python is ignoring the decimal places. Why?
Python makes a distinction between integer data and real (floating point) data.
This is because, in hardware, integers are faster to work with than fractions and other non-integers.

So when we call:

```
volume_of_pyramid(1,1)
```

Python computes 1**2*1/3
=1/3
Because Python is dividing two integers, Python ensures an integer result by ignoring decimal places.

We should thus include in the documentation for volume_of_pyramid that it produces an integer result when given integer inputs.
However, we probably want a real result. How can we guarantee that?
Solution:

```
def volume_of_pyramid(b,h):
        return b**2*h/3.0

# Note that 3.0 doesn't look like an integer to Python
```

Then:

```
>>> volume_of_pyramid(1,1)
0.33333333333333333[etc.]
```

because 1/3.0 is floating point division.

Try:

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

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

Conditional Computations

Exercise: Compute the absolute value of a number.
Eg. |5|=5, |-5|=5, |0|=0

|x|= x if x≥0
= -x if x<0

Testing conditionals in Python:

```
>>> 5>0
True

>>> 5<0
False
```

Comparison operators (>, <, etc) return True or False.
True/False represent a new type of data - Boolean values.

```
>>> x= 5>0
>>> x=True
>>> type (5>0)
<type 'bool'>
```

Other comparison operators:
greater than or equal (≥) is this in Python: >=
less than or equal (≤) is this in Python: <=
equal (=) is this in Python: ==
not equal (≠) is this in Python: !=

```
# Is 5 (not equal) to 5?
>>>5 != 5
False
# So 5 is equal to 5.
```

What can we do with Boolean values?
Make decisions based on them!
Absolute value: Test whether parameter ≥ 0. Act on that test.

```
# Absolute value function, here we go.

def abs(x):
        if x >= 0:
                return x
        else:
                return -x

>>> abs(5)
5

>>> abs(-5)
5

>>> abs(0)
0
```

Note the new piece of syntax! IF/ELSE

General format of an "if" construction:

```
        if <boolean test>:
                <true - part>
```

```
    else:
        <false - part>

# Evaluates the boolean test.
# If the test is true, then it executes the true part.
# If the test is false, it executes the false part.
```

Another example:

Exercise:
Convert numeric grades into letter grades.

90+	A+
80-89	A
70-79	B
60-69	C
50-59	D
< 50	F

Let's get started.

```
# numeric should be in the range 0-100

def letter_grade(numeric):
    if numeric >= 90:
        return "A+"
    else:
        if numeric >= 80:
            return "A"
        else:
            if numeric >= 70:
                return "B"
            else:
                if numeric >= 60:
                    return "C"
                else:
                    if numeric >= 50:
                        return "D"
                    else:
                        return "F"
```

Note that everything from "if numeric >= 80:" on down is part of the first "else".
****Python uses indentation to figure out how statements are grouped. Everything that

is part of the "else" must be indented.****
When there are more than two options among which to select, there is a more compact construction:

#Same program, but much more compact to stop from running off the page.

```
def letter_grade(numeric):
        if numeric >= 90:
                return "A+"
        elif numeric >= 80:
                return "A"
        elif numeric >= 70:
                return "B"
        elif numeric >= 60:
                return "C"
        elif numeric >= 50:
                return "D"
        else:
                return "F"
```

What did we learn today?
- Floating points
- Booleans
-Conditionals
-If/Elif/Else
-ASSIGNMENT DUE NEXT FRIDAY SEPT. 25

-Notes by Ethan Macaulay