Introduction to Python Programming

Programming

- Programs are written by humans to perform specific tasks
- Some programming languages like Python can be written in a form that is readable to humans
- Python then converts this code that is readable to humans to code that is readable to machines

Compilers

- Compilers convert human readable programs in machine readable format

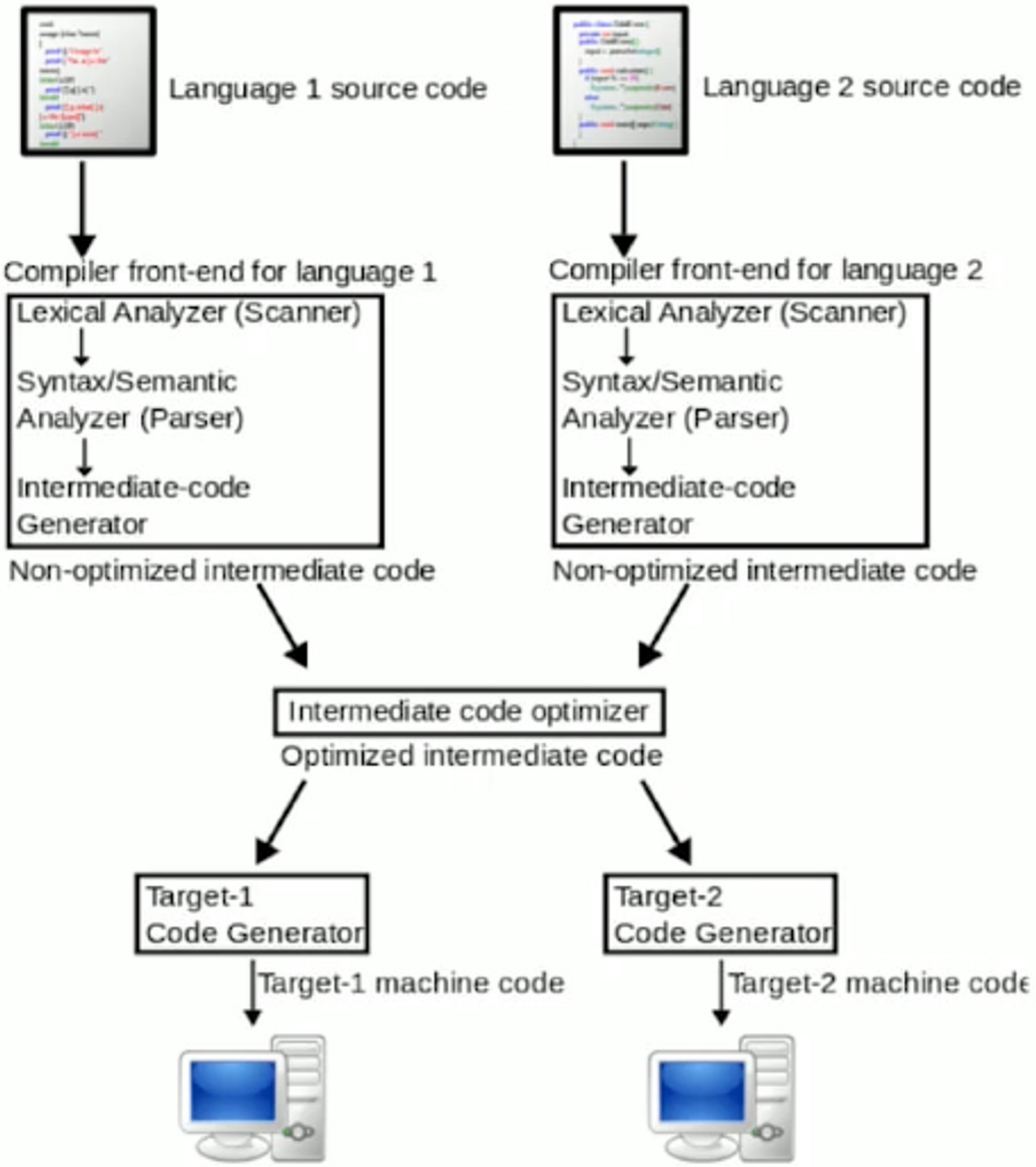
```
function my_func(foo, bar) {
   var bud = foo;
   if (bar && typeof bar ===
   "string") {
        bar = bar + "blah";
   }
   return bud + bar;
};
```

Human Readable Code

```
section .textglobal _start,
   writewrite:
            mov     al, 1 ;
   write syscall
            syscall
            ret
   _start:
            mov     rax,
   0x0a68732f6e69622f
         push    rax
            xor     rax, rax
            mov     rsi, rsp
            …
```

Machine Code

Compiler



Source Wiki

Syntax

- Compiler needs the human written code to follow rules on how the symbols are organized to form code

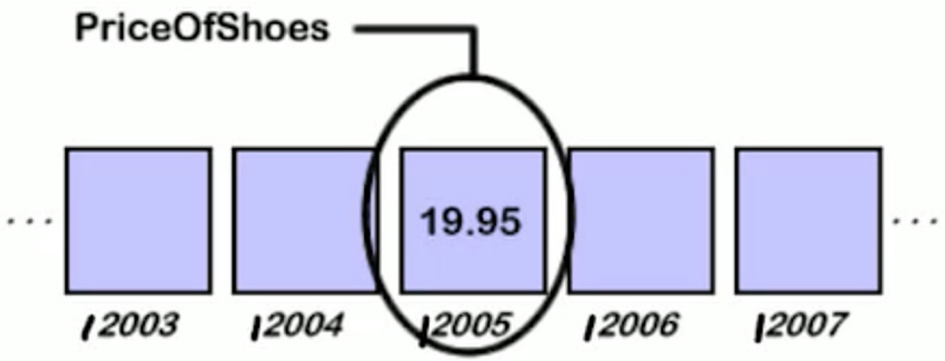
- In Perl
  - sub defines a function
- In Java
  - sub is just a plain text

```
def add5(x):
   return x+5

def dotwrite(ast):
   nodename = getNodename()
   label=symbol.sym_name.get(int(ast[0]),ast[0])
   print '    %s [label="%s' % (nodename, label),
   if isinstance(ast[1], str):
      if ast[1].strip():
         print '= %s"];' % ast[1]
      else:
         print '"]'
   else:
      print '"];'
      children = []
      for  in n, childenumerate(ast[1:]):
         children.append(dotwrite(child))
      print ,'    %s -> {' % nodename
      for  in :namechildren
         print '%s' % name,
```

Variables

Variable

- Identifiers are names we provide that are descriptive and are associated with a location in the memory
- Variables are these identifiers which has a value stored in the associated memory location
  - Value stored in the associated memory address may change



Variable in Python

```
x=3
y=6


z=x+y
print(z)
```

Data Types

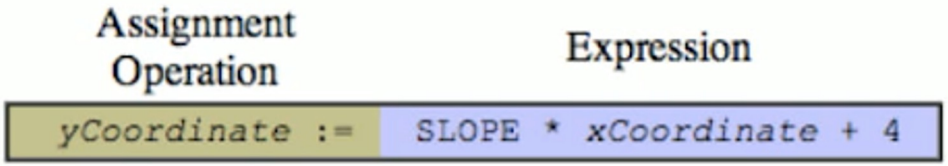
- We also need to tell what kind of values can be stored in the variable

- Strongly typed languages have explicitly defined data types such as

- Integer - 5
- Double – 5.0
- String - five

Expressions

- Combination of variables and constants assigned to another variable

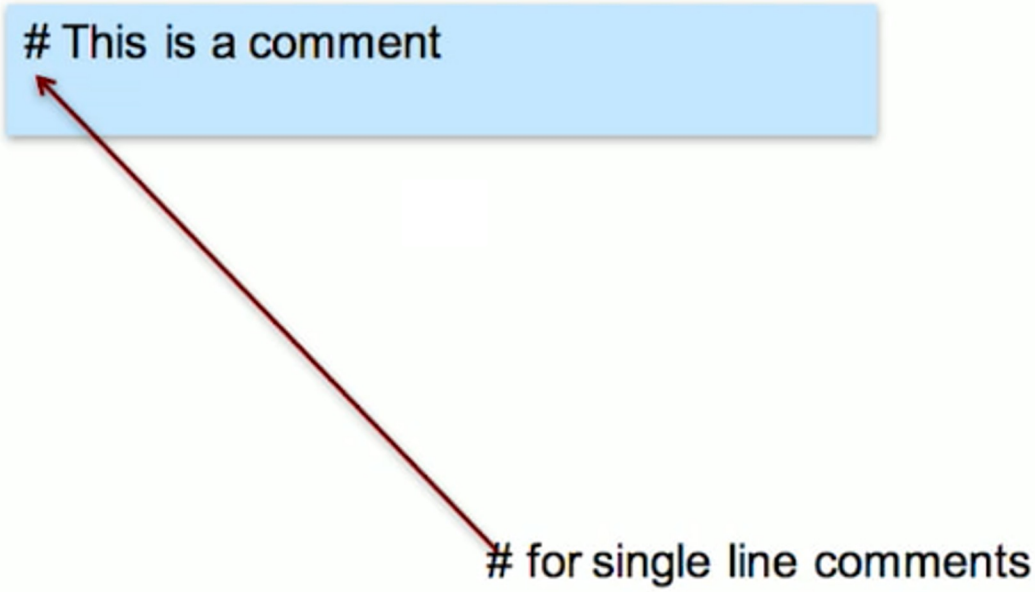


- Order of operation pre-defined

```
result := (6 + 10) / 2 - 4 * 2
       := 16 / 2 - 8
       := 8 - 8
       := 0
```

Comments

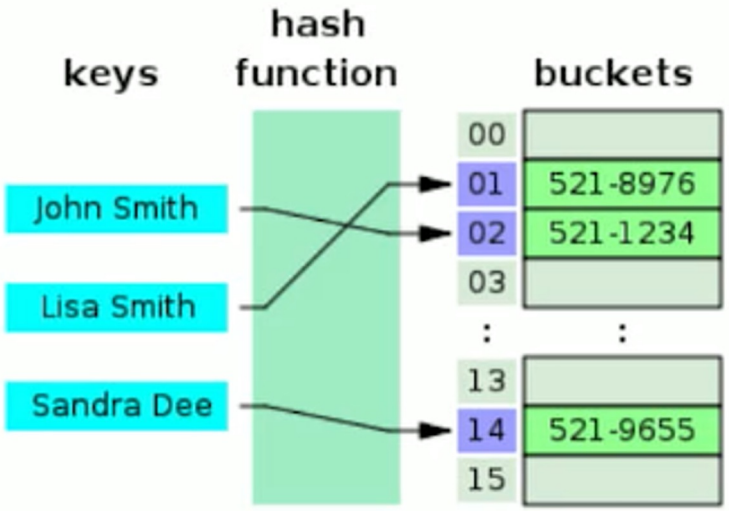
- All programming languages have unique ways to add comments



Data Structures

Data Structures

- We can assemble data in ways that allow us to process the data more efficiently
- Depending on the application different kinds of data structures are useful in different cases



Source - wiki

Lists

- Lists allows us to define one variable that can store many other variables

```
#empty list
my_list = []

#list of strings
student_list = ["sam", "alice", "jon"]

#list of integers
score_list = [130, 140, 150]

print(student_list[2])
```

Dictionary

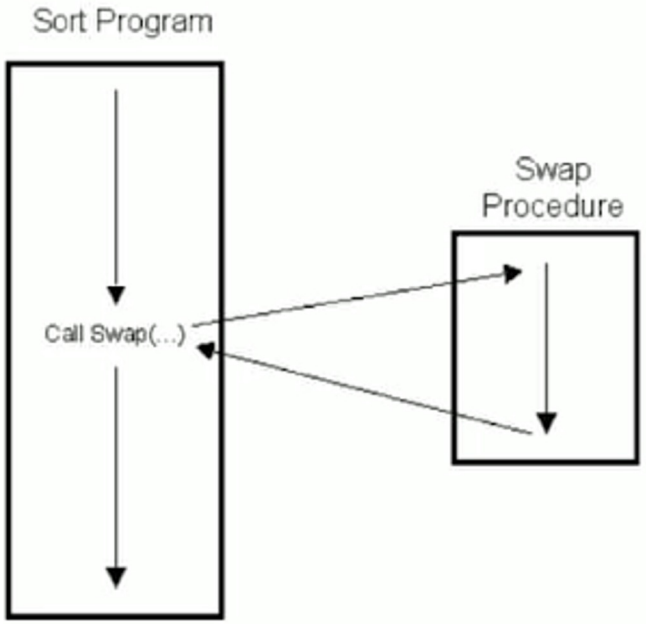
- Dictionaries allow a variable to store several key value pairs
- Very useful as a look up table

```
#empty dictionary
my_dict = {}

#student_grades
student_grades = {
    "sam":130,
    "alice":140,
    "jon":150}

print(student_grades["jon"])
```

Functions

Functions

- Functions are subprograms that can execute several expression that are grouped together to produce an output



picture source : web

Functions

```
def get_average(scores):
  i=0
  sum=0
  tot_students = len(scores)
  while (i < tot_students):
    sum=sum+scores[i]
    i=i+1
  average = sum/tot_students
  return average
```

Function Name

Function Parameter

Function Return Value

```
grades = [80,92, 75, 34, 99]

avg_grade = get_average(grades)

print(avg_grade)
```

Conditionals

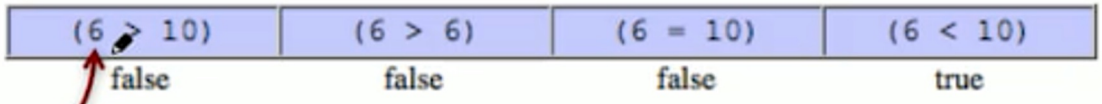
Conditional Code

- If-then-else construct is very common way of adding conditions in the code

```
IF (boolean condition){
    //consequent
}
ELSE{
   //other alternative
}
END IF
```

If Statement

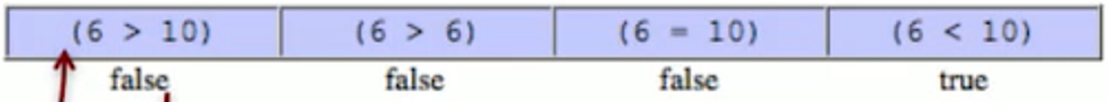
- If statement checks a boolean condition




```
IF (boolean condition){
    //consequent
}
ELSE{
   //other alternative
}
END IF
```

Else Statement

- If the boolean condition is false, the actions provided by else statement is executed




```
IF (boolean condition){
    //consequent
}
ELSE{
   //other alternative
}
END IF
```

```
x=6
if (x > 10):
  print "value is greater"
else:
   print "value is lower"
```

Boolean Expressions

AND operator

Boolean Expression	Truth Value
(T) AND (T)	T
(T) AND (F)	F
(F) AND (T)	F
(F) AND (F)	F

OR operator

Boolean Expression	Truth Value
(T) OR (T)	T
(T) OR (F)	T
(F) OR (T)	T
(F) OR (F)	F

Source - http://courses.cs.vt.edu/~csonline/ProgrammingLanguages/Lessons/Booleans/index.html

Loops

Loops

- Loops are very important constructs in programming language because it allows the computer to run the same expression repeatedly

- Imagine you are taking the average of grades for all students in a class

Iteration Concept

- We can have a piece of code run any number of times

```
i=0
while i < 10:
  print(i)
  i=i+1
```

While Loop

- While loop provides iteration construction with boolean conditional

```
scores = [80,92, 75, 34, 99]
i=0
sum=0
tot_students = len(scores)
while (i < tot_students):
  sum=sum+scores[i]
  i=i+1
average = sum/tot_students
print(average)
```

```
cities = ["New York City", "Boston", "Kathmandu", "Kigali"]
for city in cities:
  print(city)
```

For Loop

- For loop is another method of writing the iteration construct

```
scores = [80,92, 75, 34, 99]
tot_students = len(scores)
for x in range(0, tot_students):
    print(scores[x])
```