UNIVERSITY OF TEXAS ARLINGTON

PHYS 5319-001:

Math Methods in Physics III

Instructor:	Dr. Qiming Zhang
Office:	CPB 336
Phone:	817-272-2020
Email:	zhang@uta.edu

Basic Concepts of Programming

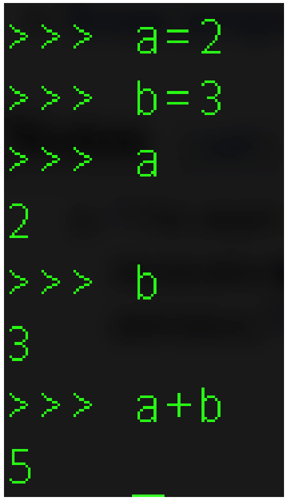
- Variables
- Data Types
- Control Structures
- Function
- Syntax
- Indentation
- Library

What is a variable?

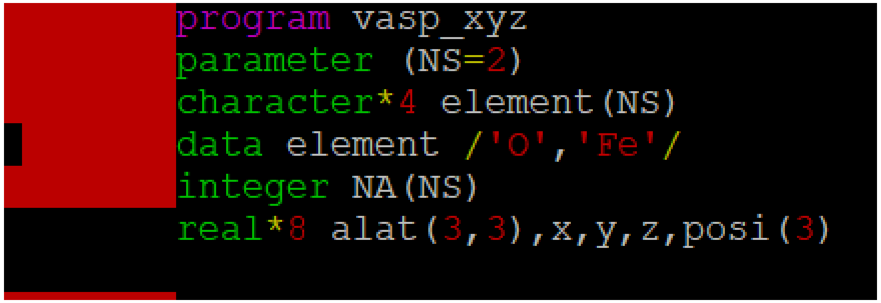
Variables are the backbone of any programming language.
In computer programming, a variable is a storage location and an associated symbolic name which contains some known or unknown quantity or information, a value.

In Python and Matlab, there is no command to declare a variable. Instead, it is created the moment the value was assigned to it.

For example, we can define two variables, ‘a’ and ‘b’, and assign them values 2 and 3. It can be realized as follows in Python:



```
>>> a=2
>>> b=3
>>> a
2
>>> b
3
>>> a+b
5
```



```
program vasp_xyz
parameter (NS=2)
character*4 element(NS)
data element /'O','Fe'/
integer NA(NS)
real*8 alat(3,3),x,y,z,posi(3)
```

In contrast, in some other language, e.g., FORTRAN, you can declare a variable first and assign a value to it later.

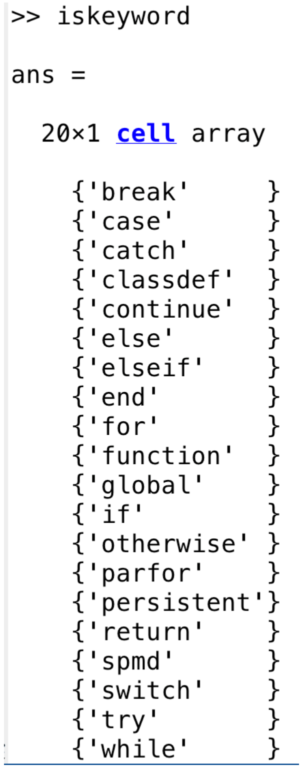
Valid variable names

- In Python
  - Start with a letter or the underscore character
- In Matlab and FORTRAN
  - starts with a letter,
- For both of them
  - can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
  - case-sensitive (age, Age and AGE are three different variables)
  - cannot define variables with the same names as Python/Matlab **keywords**

For example, Matlab keywords:

Keywords are the reserved words in Matlab. They are case sensitive.
Similar (not exact) in FORTRAN.

Matlab



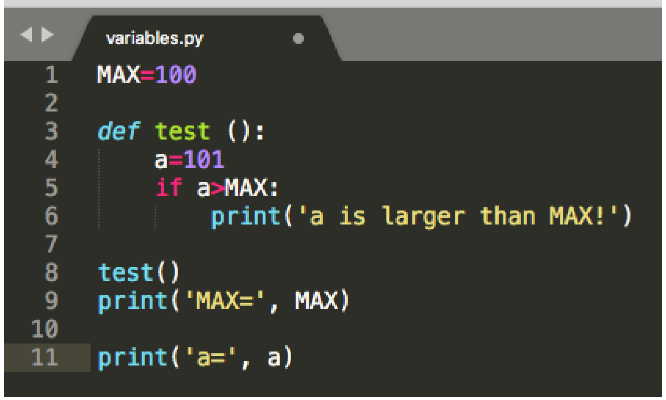
```
>> iskeyword

ans =

  20×1 cell array

    {'break'     }
    {'case'      }
    {'catch'     }
    {'classdef'  }
    {'continue'  }
    {'else'      }
    {'elseif'    }
    {'end'       }
    {'for'       }
    {'function'  }
    {'global'    }
    {'if'        }
    {'otherwise' }
    {'parfor'    }
    {'persistent'}
    {'return'    }
    {'spmd'      }
    {'switch'    }
    {'try'       }
    {'while'     }
```

Global vs. local variables

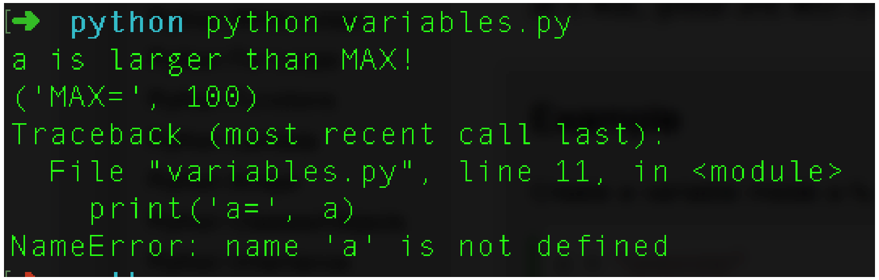
Variables that are created outside of a function are known as global variables.
Global variables can be used by everyone, both inside of functions and outside.

If a variable is created inside a function, it will be local, and can only be used inside the function.

For example, variable ‘a’ is defined within function ‘test()’. It is local, it cannot be used outside of ‘test()’ function. Variable ‘MAX’ is defined out of all functions. It is global, and can be used both inside and outside of functions.



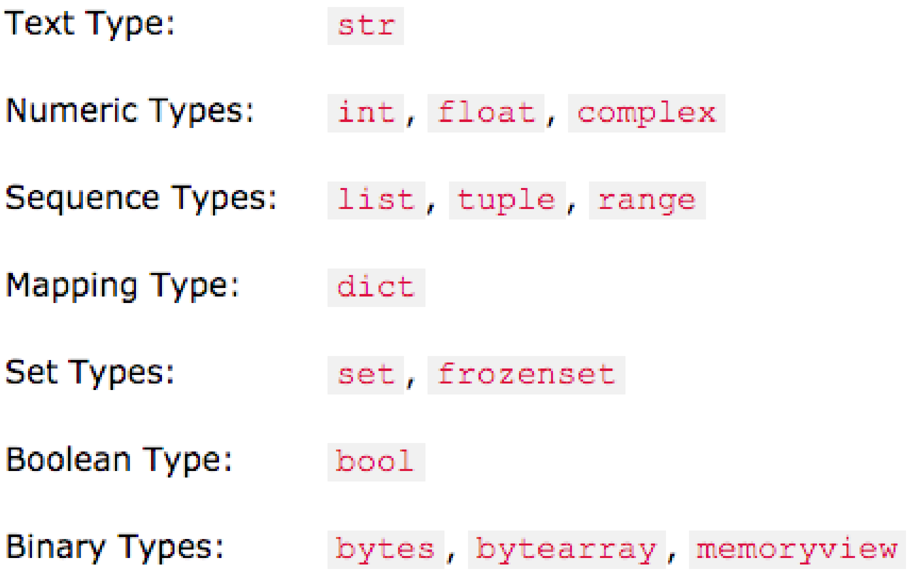
```python
MAX=100

def test ():
    a=101
    if a>MAX:
        print('a is larger than MAX!')

test()
print('MAX=', MAX)

print('a=', a)
```



```
➜  python python variables.py
a is larger than MAX!
('MAX=', 100)
Traceback (most recent call last):
  File "variables.py", line 11, in <module>
    print('a=', a)
NameError: name 'a' is not defined
```

Data types

Variables can store data of different types, and different types can do different things.
e.g. Python has the following data types built-in by default, in these categories:



Text Type:	`str`
Numeric Types:	`int`, `float`, `complex`
Sequence Types:	`list`, `tuple`, `range`
Mapping Type:	`dict`
Set Types:	`set`, `frozenset`
Boolean Type:	`bool`
Binary Types:	`bytes`, `bytearray`, `memoryview`

Number representation

Binary: unit 1 bit; possible value: 0 or 1

2 (decimal) → 00000010 (binary)
octal, hex

For N bit, only 2^N integers can be represented, take 1 bit for sign (+/-), the possible value: $[0, 2^{N-1}]$

Word length (~precision): 8-bit, 32-bit ($2^{31} \approx 2 \times 10^9$), 64-bit

1 byte (1B)=8 bits; e.g. 64 MB
1 K→1024 (=2^{10})

Usually, 1B→single character. ASCII characters has 256 (=2^8) characters

$[0, 255]$

Real number

- In floating-point notation (preferred scientifically)

To represent a *x*, we need: sign, mantissa & exponential_field

$x_{\text{float}} = (-1)^{s} \times \text{mantissa} \times 2^{\text{expfld-bias}}$

e.g. A 32-bit machine (real*4 in FORTRAN, short(Matlab?)):

8 bit for expfld [0, 255] → [-127, 128] ($2^{128} \approx 10^{38}$)

(*watch for overflow/underflow errors*)

remaining: 1 for sign, 23 for the mantissa (6-7 decimals)

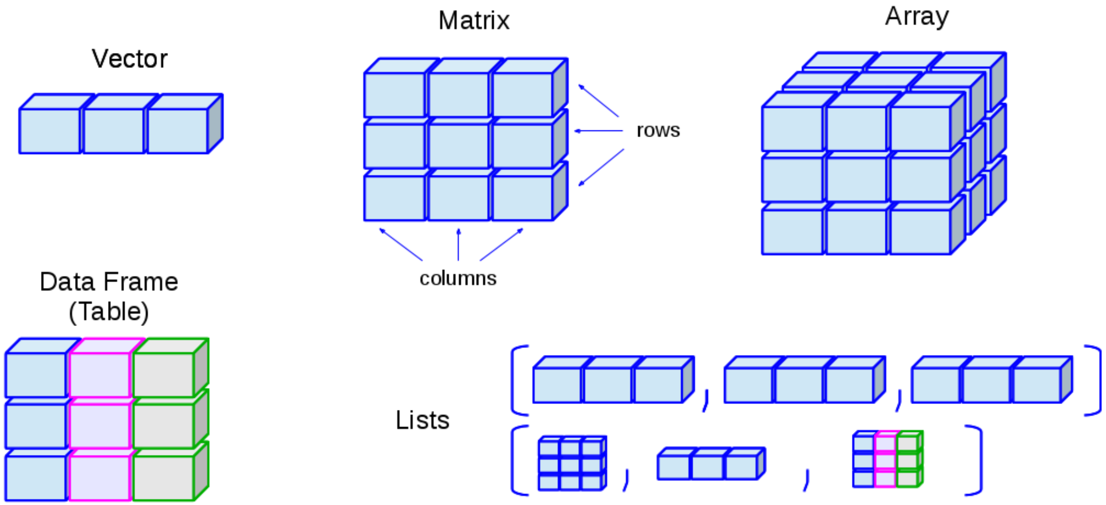
- Double precision (real*8 in FORTRAN, long(Matlab?)):

  64-bit: 1 for S, 11 bits expfid, 52 bits mantissa (~15 decimal)

  - *Exercise: print 2 real numbers of 1./3. to see their decimals (32-bit/64-bit)*

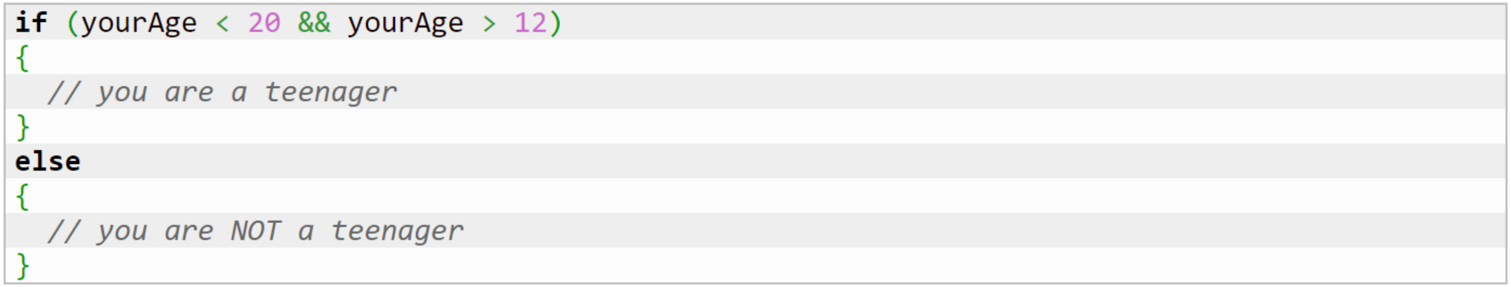
✓ Data Structures

A data structure is a particular way of storing and organizing data in a computer so that it can be used efficiently.

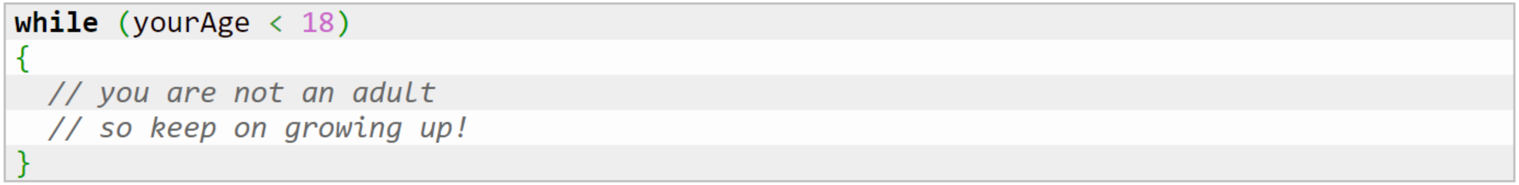


What is a control structure?

A control structure is a block of programming that analyzes variables and chooses a direction in which to go based on given parameters. The term flow control details the direction the program takes (which way program control "flows"). Hence it is the basic decision-making process in computing; flow control determines how a computer will respond when given certain conditions and parameters.

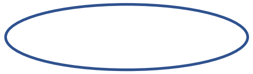


```
if (yourAge < 20 && yourAge > 12)
{
  // you are a teenager
}
else
{
  // you are NOT a teenager
}
```

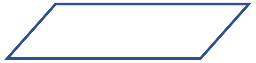


```
while (yourAge < 18)
{
  // you are not an adult
  // so keep on growing up!
}
```

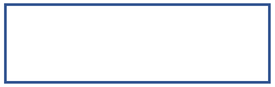
Structured programming (flowchart)



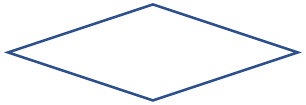
Beginning/end of the algorithm



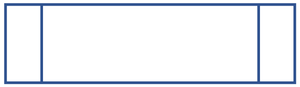
input/output



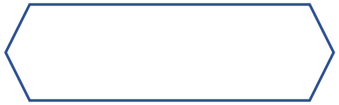
execution(s)



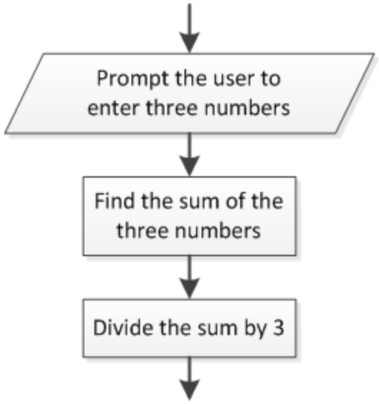
branching point (logical operation)

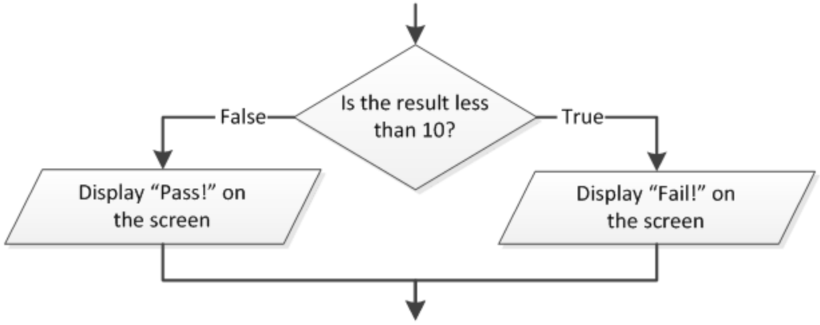


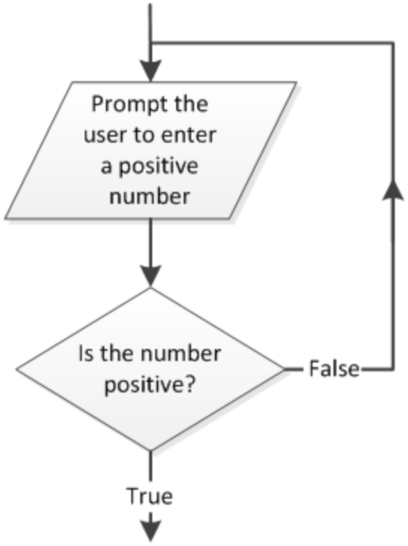
a sub-algorithm whose exact structure explained elsewhere



loop

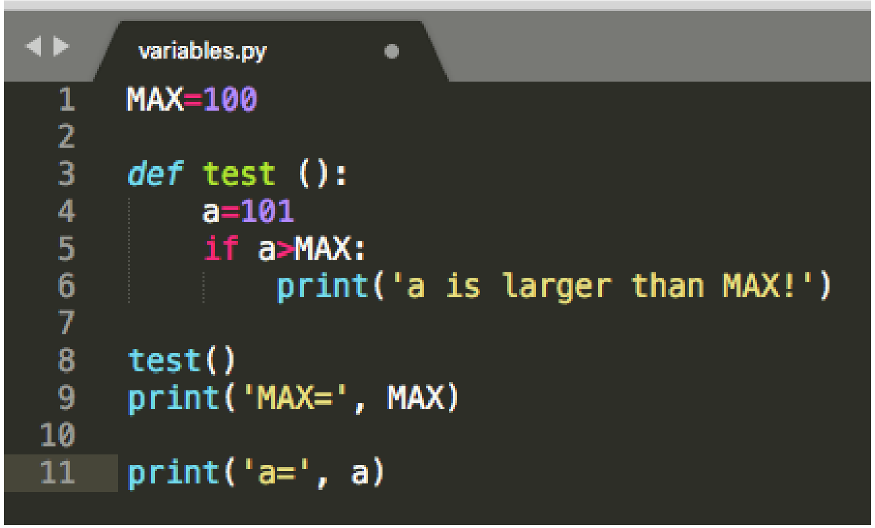

Prompt the user to enter three numbers
Find the sum of the three numbers
Divide the sum by 3


Is the result less than 10?
False
True
Display "Pass!" on the screen
Display "Fail!" on the screen



Function

A function is a block of code which only runs when it is called.
You can pass data, known as parameters, into a function.
A function can return data as a result.

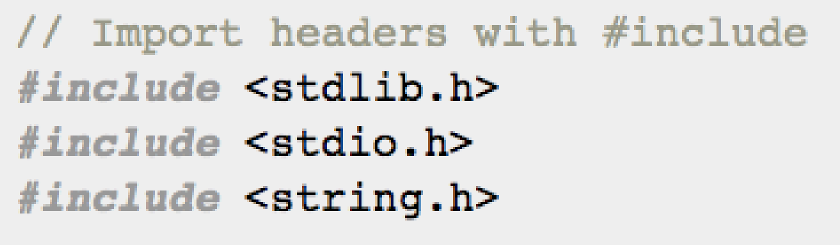


```python
MAX=100

def test ():
    a=101
    if a>MAX:
        print('a is larger than MAX!')

test()
print('MAX=', MAX)

print('a=', a)
```

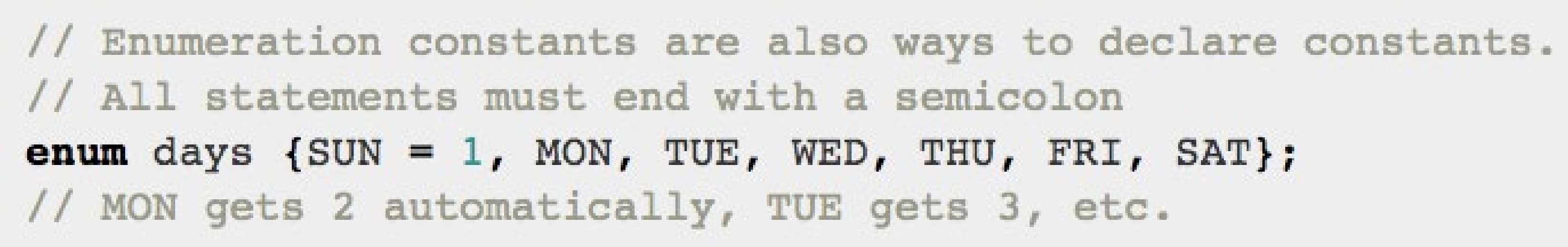
syntax

In computer science, the syntax of a programming language is the set of rules that define the combinations of symbols that are considered to be correctly structured programs in that language.

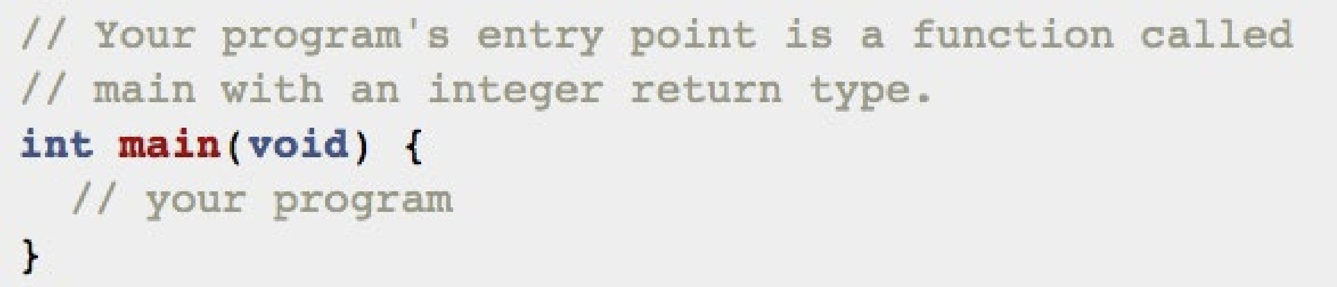
For example, in C https://learnxinyminutes.com/



```c
// Import headers with #include
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
```

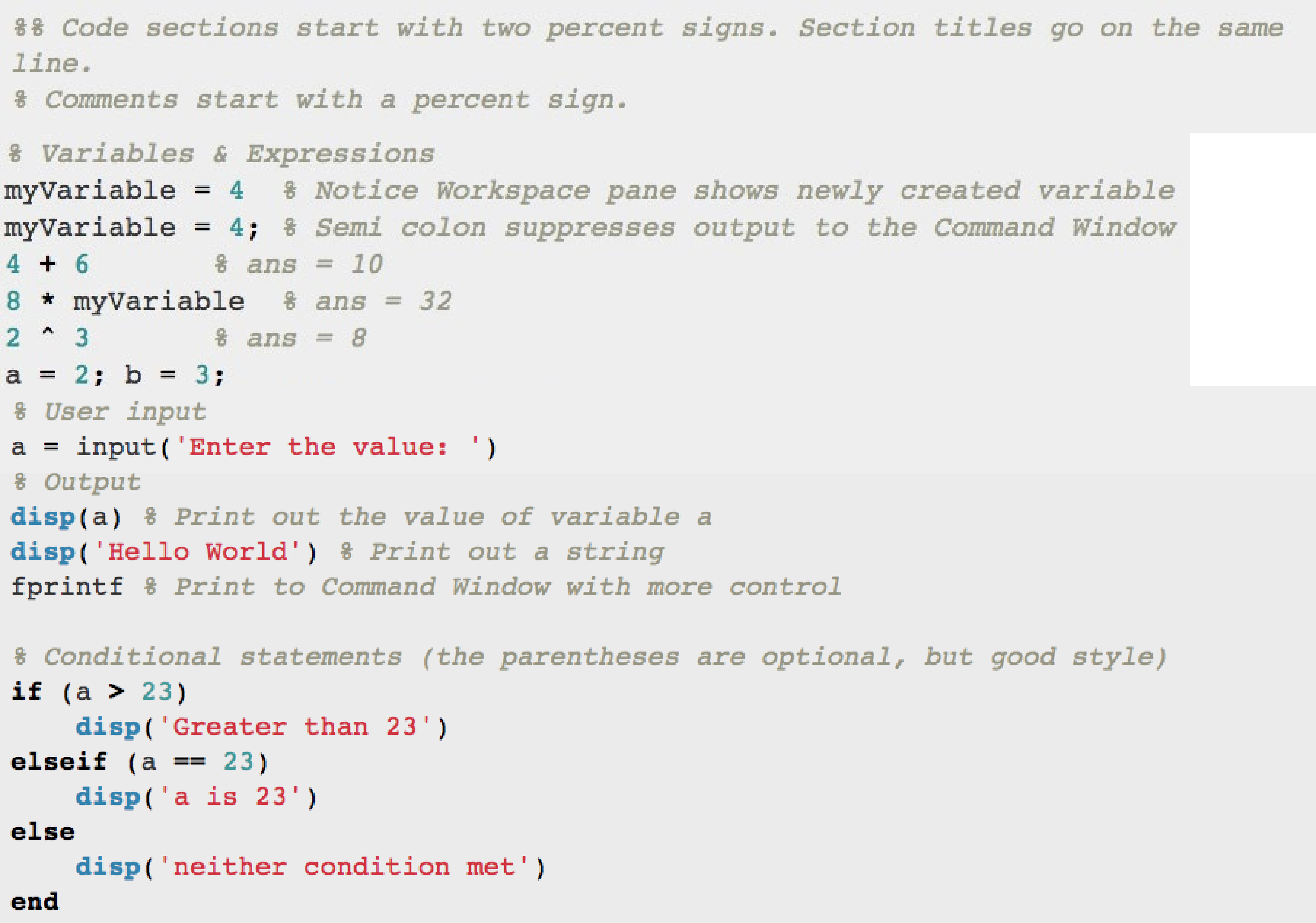


```c
// Enumeration constants are also ways to declare constants.
// All statements must end with a semicolon
enum days {SUN = 1, MON, TUE, WED, THU, FRI, SAT};
// MON gets 2 automatically, TUE gets 3, etc.
```



```c
// Your program's entry point is a function called
// main with an integer return type.
int main(void) {
  // your program
}
```

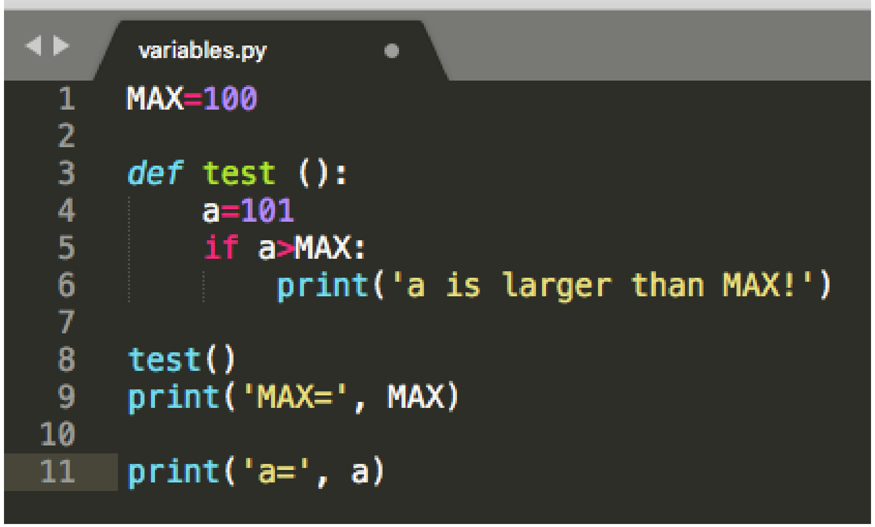
in Matlab



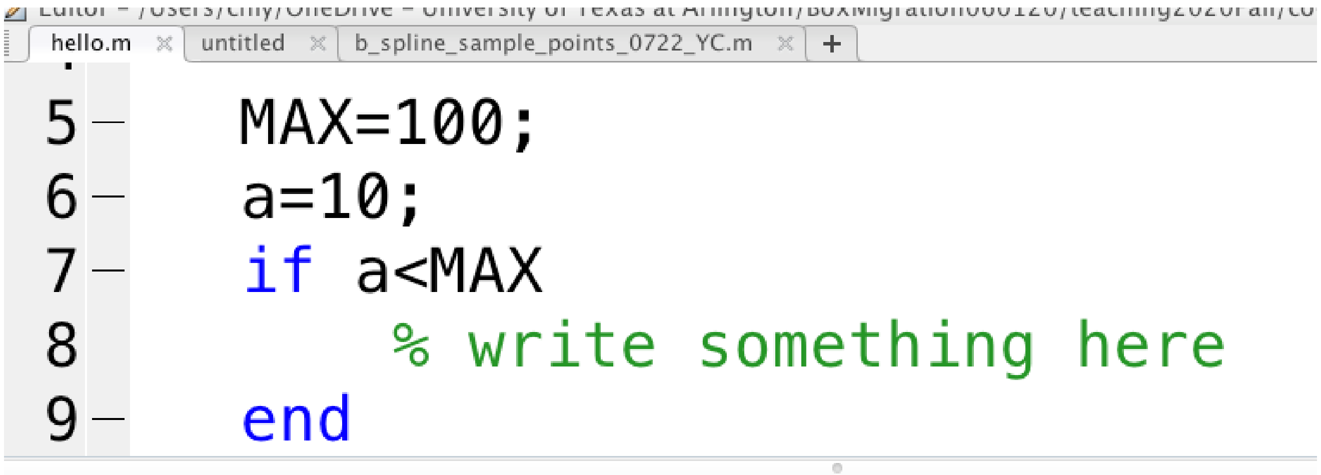
```
%% Code sections start with two percent signs. Section titles go on the same
line.
% Comments start with a percent sign.

% Variables & Expressions
myVariable = 4  % Notice Workspace pane shows newly created variable
myVariable = 4; % Semi colon suppresses output to the Command Window
4 + 6       % ans = 10
8 * myVariable  % ans = 32
2 ^ 3       % ans = 8
a = 2; b = 3;
% User input
a = input('Enter the value: ')
% Output
disp(a) % Print out the value of variable a
disp('Hello World') % Print out a string
fprintf % Print to Command Window with more control

% Conditional statements (the parentheses are optional, but good style)
if (a > 23)
    disp('Greater than 23')
elseif (a == 23)
    disp('a is 23')
else
    disp('neither condition met')
end
```

Indentation

Python uses indentation for blocks, instead of curly braces. Both tabs and spaces are supported, but the standard indentation requires standard Python code to use four spaces.



```
MAX=100

def test ():
    a=101
    if a>MAX:
        print('a is larger than MAX!')

test()
print('MAX=', MAX)

print('a=', a)
```



```
MAX=100;
a=10;
if a<MAX
    % write something here
end
```

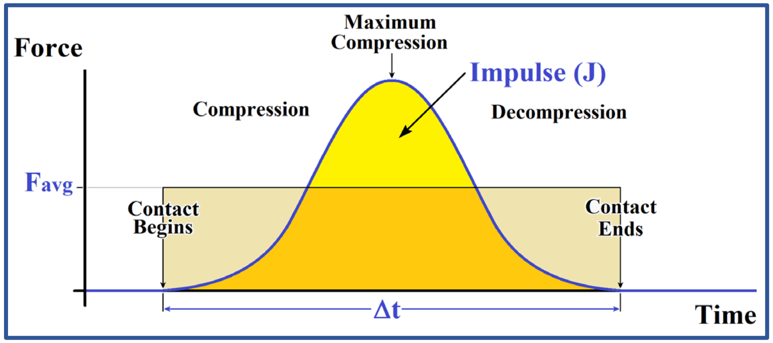
Library

In computer science, a library is a collection of non-volatile resources used by computer programs, often for software development. These may include configuration data, documentation, help data, message templates, pre-written
code and subroutines, classes, values or type specifications.

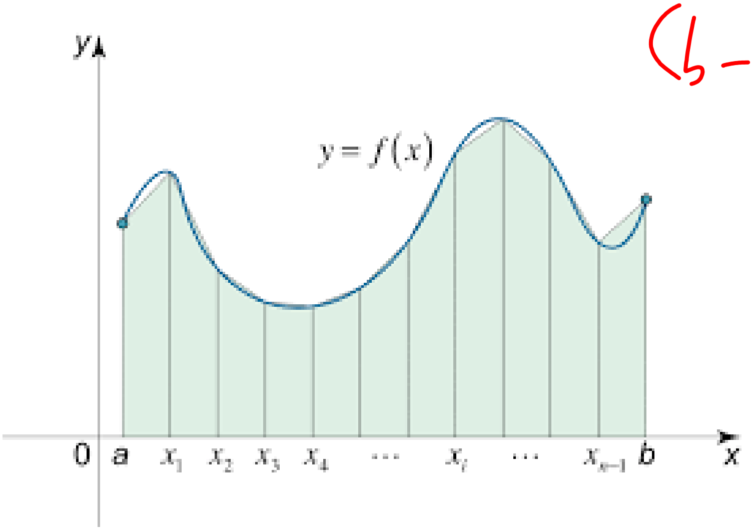
For example: Linear Algebra package: LaPack. Scalapack (parallel version)
www.netlib.org/lapack/

Numerical integration

$$\vec{J} = \int_{t_1}^{t_2} \vec{F}(t)dt$$



$(b-a)f(\xi)$



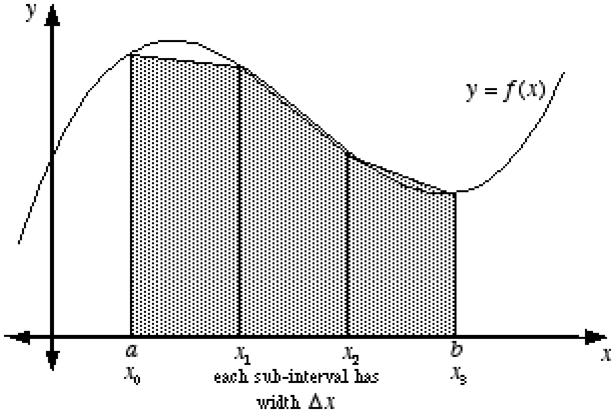
$$\int_a^b f(x)dx \approx \sum_{i=1}^{N} f(x_i)w_i$$

where N: # of subintervals
w_i : sum weight for *i-th* subinterval
x_i: a point in *i-th* subinterval

How to choose points/weights?

Newton-Cotes methods

f(x) ~ a few terms in a Taylor expansion of *f*;
[a, b] interval is divided into equal subintervals.



The area of the trapezoids (shaded) approximately equals the area bounded by $y = f(x)$.

$$\int_a^b f(x)dx = \frac{\Delta x}{2}[f(x_0) + 2f(x_1) + 2f(x_2) + f(x_3)].$$

-Trapezoid Rule:

Subinterval (totally N-1) $h = \frac{b-a}{N-1}$, ***N*: arbitrary integer**

$$x_i = a + (i-1)h, i = 1, 2, \ldots, N$$

$$f(x) \approx \alpha x + \beta \qquad \text{(a straight line)}$$

2 parameters (α, β) to be determined. Need 2 points: *i, i+1:*

$$\int_{x_i}^{x_{i+1}} f(x)dx \approx h\frac{f_i + f_{i+1}}{2} = \frac{1}{2}hf_i + \frac{1}{2}hf_{i+1} \qquad O(h^3 f'')$$

Apply to the entire region [a,b]:

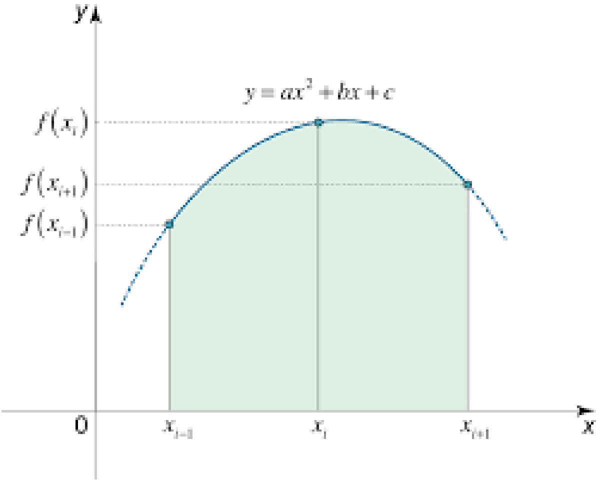
$$\int_a^b f(x)dx = \frac{h}{2}f_1 + hf_2 + hf_3 + \cdots + hf_{N-1} + \frac{h}{2}f_N + O(h^2 f'')$$

-Simpson rule

Subinterval (totally N-1) $\boldsymbol{h = \frac{b-a}{N-1}}$**,** ***N*: odd integer**

$$\boldsymbol{x_i = a + (i-1)h, i = 1, 2, \ldots, N}$$

$$\boldsymbol{f(x) \approx \alpha x^2 + \beta x + \gamma} \text{ (a quadratic line)}$$



3 parameters (α, β, γ) to be determined. Need 3 points: *i, i-1, & i+1:* *(take h=1)*

$$\boldsymbol{\int_{-1}^{+1} (\alpha x^2 + \beta x + \gamma) dx = \frac{2}{3}\alpha + 2\gamma}$$

But $\boldsymbol{f(-1) = \alpha - \beta + \gamma}$

$\boldsymbol{f(0) = \gamma}$

$\boldsymbol{f(1) = \alpha + \beta + \gamma}$

So $\boldsymbol{\int_{-1}^{+1} (\alpha x^2 + \beta x + \gamma) dx = \frac{1}{3} f(-1) + \frac{4}{3} f(0) + \frac{1}{3} f(1)}$

Or $\boldsymbol{= \frac{h}{3} f_{i-1} + \frac{4h}{3} f_i + \frac{h}{3} f_{i+1}}$

$$\int_a^b f(x) dx = \frac{h}{3} f_1 + \frac{4h}{3} f_2 + \frac{2h}{3} f_3 + \cdots + \frac{4h}{3} f_{N-1} + \frac{h}{3} f_N + O(h^4)$$