Paradigms in programming languages

Bikram Shrestha
ECE 492

What is programming language?

- A programming language is a notational system for describing computing tasks in both a machine- and human-readable form.

- Every programming language has its own notation

http://homepage.cs.uiowa.edu/~slonnegr/plf/Book/Chapter6.pdf

Paradigms

- Procedural and Structured Programming
- Functional programming
- Logic programming
- Object-oriented programming
- Other Paradigms

Procedural programming

- change of program state as function of time.
- series of command to execute to complete a task.
- Languages: Fortran, Algol, Pascal, Basic C

```
/* C basic structure program Documentation section
Author: fresh2refresh.com
Date : 01/01/2012
*/#include <stdio.h> /* Link section */
int total = 0; /* Global declaration and definition section */
int sum (int, int); /* Function declaration section */
int main () /* Main function */
{
printf ("This is a C basic program \n");
total = sum (1, 1);
printf ("Sum of two numbers : %d \n", total);
return 0;
}
int sum (int a, int b) /* User defined function */
{ /* definition section */
return a + b;
}
```

LAFAYETTE
ELECTRICAL & COMPUTER ENGINEERING

https://fresh2refresh.com/c/c-basic-program/

Functional Programming

- Provide functions to do calculation.
- Mathematics and theory of functions.
- Functions as first class object.
- No state information.
- Languages : LISP

```
(define (factorial n)
    (cond   ((equal? n 0) 1)
            (#t (* n (factorial (- n 1))))))
```

LAFAYETTE
ELECTRICAL & COMPUTER ENGINEERING

http://homepage.cs.uiowa.edu/~slonnegr/plf/Book/Chapter6.pdf

Logic programming

- Based on inference rules, axioms and queries
- Search in for set of facts or use of set of inference rules.
- Language : Prolog

```
sibling(X,Y) :- parent(Z,X), parent(Z,Y)
parent(X,Y) :- father(X,Y)
parent(X,Y) :- mother(X,Y)
mother(megan, brian).
father(ryan, brian)
father(ryan, molly)
father(mike, ryan)
```

Object Oriented Programming

- Data and operations encapsulated in object.
- Abstraction, encapsulation, inheritance and polymorphism
- Modular
- Real time simulation.
- Languages: C++, Java

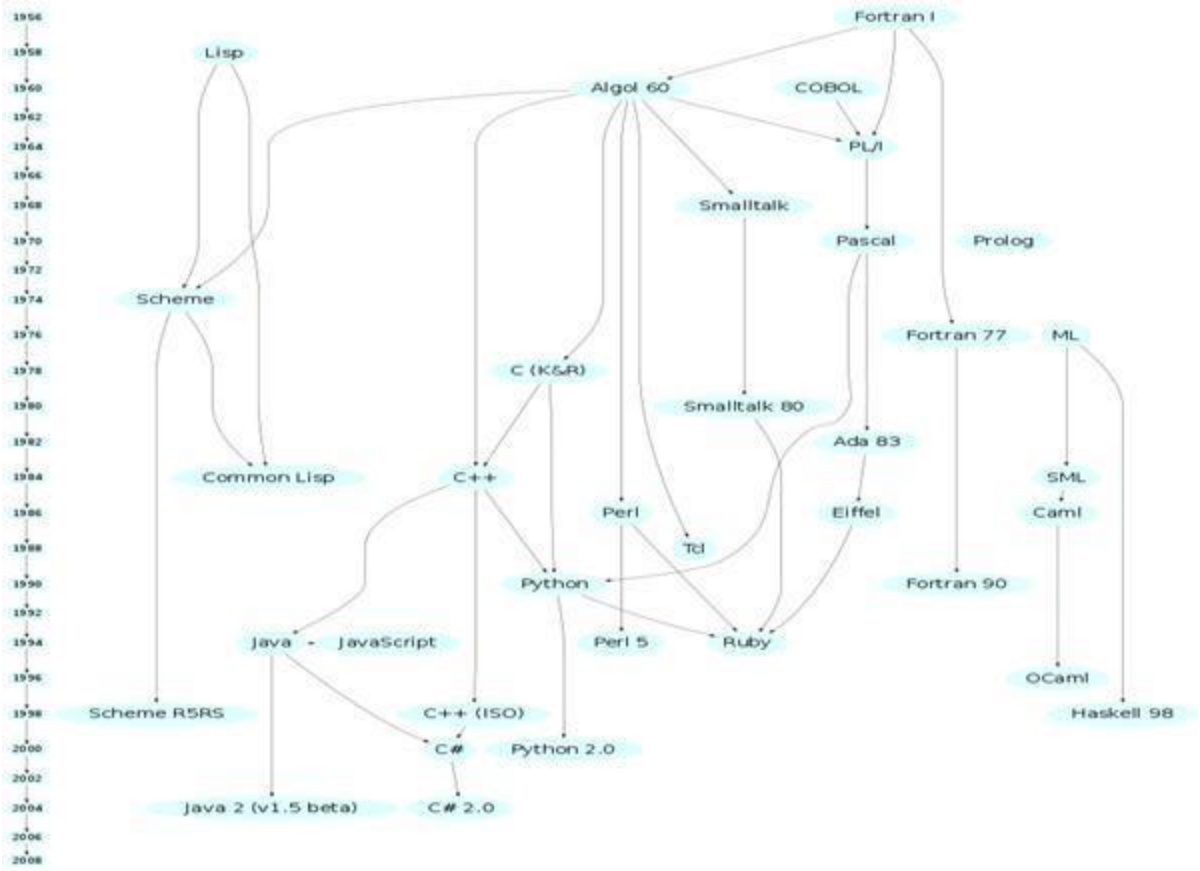
```
class Bicycle {

    int cadence = 0;
    int speed = 0;
    int gear = 1;

    void changeCadence(int newValue) {
         cadence = newValue;
    }

    void changeGear(int newValue) {
         gear = newValue;
    }

    void speedUp(int increment) {
         speed = speed + increment;
    }

    void applyBrakes(int decrement) {
         speed = speed - decrement;
    }

    void printStates() {
         System.out.println("cadence:" +
             cadence + " speed:" +
             speed + " gear:" + gear);
    }
}
```

LAFAYETTE
ELECTRICAL & COMPUTER ENGINEERING

http://docs.oracle.com/javase/tutorial/java/concepts/

Other Paradigms

- Multi-paradigm programming
  - python and ruby
- concurrent programming
  - Use of multiple threads to achieve parallelism
- Limitations of programming language in one paradigm.



LAFAYETTE
ELECTRICAL & COMPUTER ENGINEERING

The history of programming languages. Source: http://rigaux.org/language-study/diagram.html

Python and multi-paradigms programming

- Lot of Built in functions.
  - Follows both procedural and functional.
- Classes as object.
  - Follows object oriented programming.
- Limitations on programming language of one paradigm.

https://docs.python.org/2/howto/functional.html

Python Examples

```
class MyClass:
    """A simple example class"""
    i = 12345
    def f(self):
        return 'hello world'
```

```
l = [('Knights', 'Ni'), ('Monty', 'Python'), ('SPAM', 'SPAAAM')]
d = dict()
for tuple in l:
    d[tuple[0]] = tuple[1]
```

```
def sumProblem(x, y):
    sum = x + y
    sentence = 'The sum of {} and {} is {}.'.format(x, y, sum)
    print(sentence)

def main():
    sumProblem(2, 3)
    sumProblem(1234567890123, 535790269358)
    a = int(input("Enter an integer: "))
    b = int(input("Enter another integer: "))
    sumProblem(a, b)
```

LAFAYETTE
ELECTRICAL & COMPUTER ENGINEERING

https://docs.python.org/2/

References

1. Structured Programming
http://delivery.acm.org/10.1145/1250000/1243380/cb-sp-dahl.pdf?ip=139.147.30.49&id=1243380&acc=ACTIVE%20SERVICE&key=A792924B58C015C1%2EA00F0F67B1677B09%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=67195865 2&CFTOKEN=51235160&__acm__=1430913185_b67e2c795b93e168f22a8aa4690817ad

2. On Integration of Programming Paradigms
http://delivery.acm.org/10.1145/240000/234735/p309-mycroft.pdf?ip=139.147.30.49&id=234735&acc=ACTIVE%20SERVICE&key=A792924B58C015C1%2EA00F0F67B1677B09%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=67195865 2&CFTOKEN=51235160&__acm__=1430897015_6bc120d0e2ee546b7133fd02e59a54a4
3. Self Definition of Programming Language/LISP
http://homepage.cs.uiowa.edu/~slonnegr/plf/Book/Chapter6.pdf
4. Imperative functional programming
http://delivery.acm.org/10.1145/240000/234736/p312-reddy.pdf?ip=139.147.30.49&id=234736&acc=ACTIVE%20SERVICE&key=A792924B58C015C1%2EA00F0F67B1677B09%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=67195865 2&CFTOKEN=51235160&__acm__=1430922254_7082045cbbc43b70787b77e6f07d8eab
5. Conception, evolution, and application of functional programming languages
http://delivery.acm.org/10.1145/800007/72554/p359-hudak.pdf?ip=139.147.30.49&id=72554&acc=ACTIVE%20SERVICE&key=A792924B58C015C1%2EA00F0F67B1677B09%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=67195865 2&CFTOKEN=51235160&__acm__=1430922376_02425edf037fb73f404b1019162eeea6
6. Complexity and expressive power of logic programming.
ACM Comput. Surv. 33, 3 (Sep. 2001), 374-4
http://delivery.acm.org/10.1145/510000/502810/p374-dantsin.pdf?ip=139.147.30.49&id=502810&acc=ACTIVE%20SERVICE&key=A792924B58C015C1%2EA00F0F67B1677B09%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=67195865 2&CFTOKEN=51235160&__acm__=1430922473_cad0e643c329592 8428c7ae78a881fab
7. Object Oriented Concepts
http://docs.oracle.com/javase/tutorial/java/concepts/
8. Python Documentation
https://docs.python.org/3/

LAFAYETTE
ELECTRICAL & COMPUTER ENGINEERING

Questions?