

# Recursion

You will learn the definition of recursion as well as seeing how simple recursive programs work

James Tam

## What Is Recursion?

*“the determination of a succession of elements by operation on one or more preceding elements according to a rule or formula involving a finite number of steps” (Merriam-Webster online)*

James Tam

## What This Really Means

*Breaking a problem down into a series of steps. The final step is reached when some basic condition is satisfied. The solution for each step is used to solve the previous step. The solution for all the steps together form the solution to the whole problem.*

(The “Tam” translation)

James Tam

## Definition For Philosophy

*“...state of mind of the wise man; practical wisdom...”<sup>1</sup>*

*See **Metaphysics***

<sup>1</sup> The New Webster Encyclopedic Dictionary of the English Language

James Tam

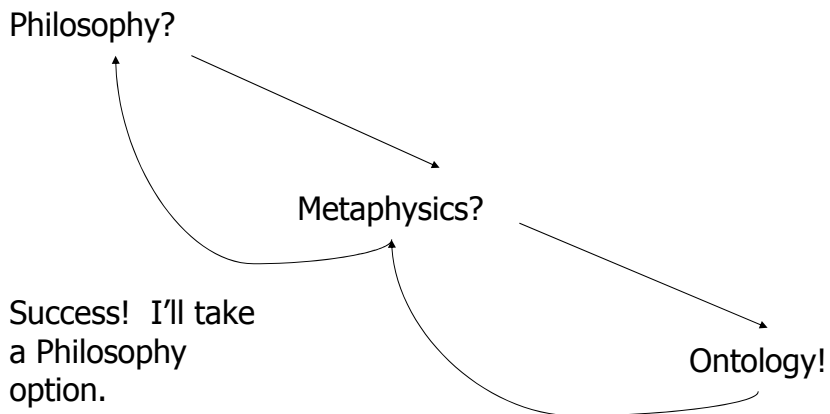
## Metaphysics

*“...know the ultimate grounds of being or what it is that really exists, embracing both psychology and **ontology**.”<sup>2</sup>*

## Result Of Lookup , Possibility One: Success

- I know what Ontology means!

## Result Of Lookup, Possibility One



James Tam

## Result Of Lookup, Possibility Two: Failure

- Lookups loop back.

James Tam

## Result Of Lookup, Possibility Two

Philosophy?

Metaphysics?

**Rats!!!**

See  
previous

Ontology?

James Tam

## Ontology

*“...equivalent to metaphysics.”<sup>3</sup>*

<sup>3</sup> The New Webster Encyclopedic Dictionary of the English Language

Wav file from "The Simpsons"

James Tam

## **Result Of Lookup, Possibility Three: Failure**

- You've looked up everything and still don't know the definition!

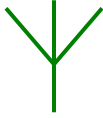
James Tam

## **Looking Up A Word**

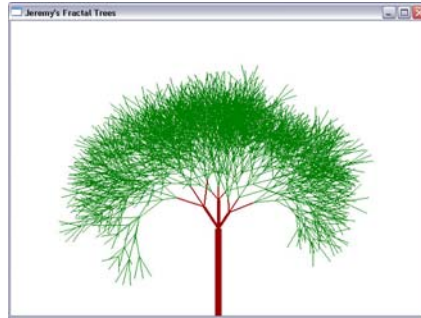
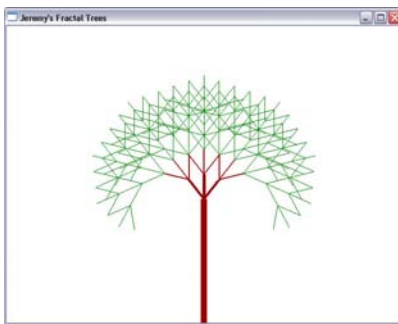
if (you completely understand a definition) then  
    return to previous definition (using the definition that's understood)  
else  
    lookup (unknown word(s))

James Tam

## Recursion: Can Be Used To Produce Graphics



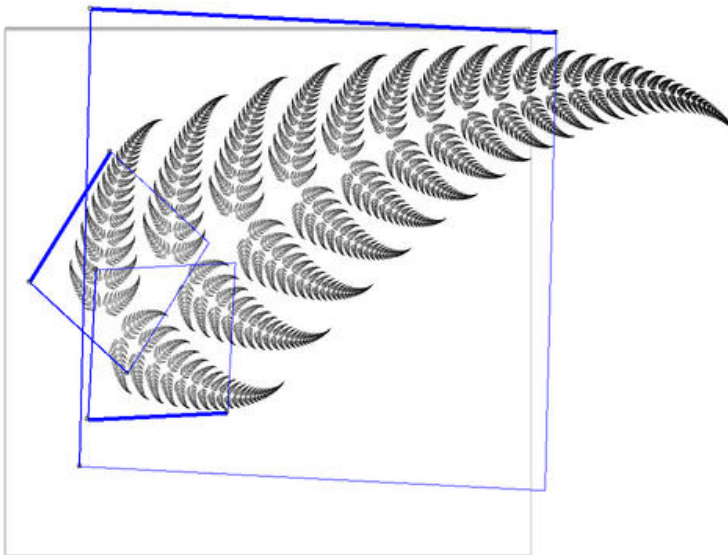
Produce a picture by repeating a pattern



Images from <http://www.csis.gvsu.edu/~marzkaj/CS367/project1.htm>

James Tam

## Recursion: Can Be Used To Produce Graphics (2)



<http://charm.cs.uiuc.edu/users/olawlor>

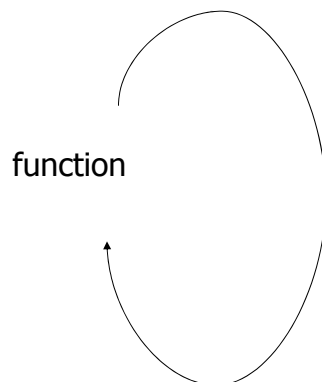
James Tam

## Recursion In Programming

*“A programming technique whereby a function calls itself either directly or indirectly.”*

James Tam

## Direct Call

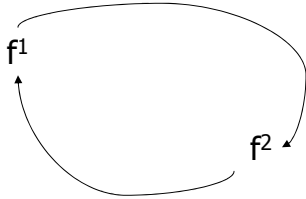


```
def fun ():  
    :  
    fun ()  
    :
```

James Tam

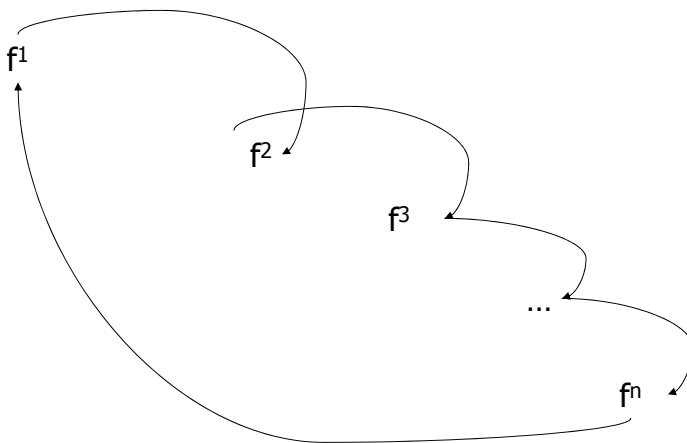


## Indirect Call



James Tam

## Indirect Call



James Tam

## **Indirect Call (2)**

Name of the example program: recursive.1py

```
def fun1 ():  
    fun2 ()
```

```
def fun2 ():  
    fun1 ()
```

```
fun1 ()
```

James Tam

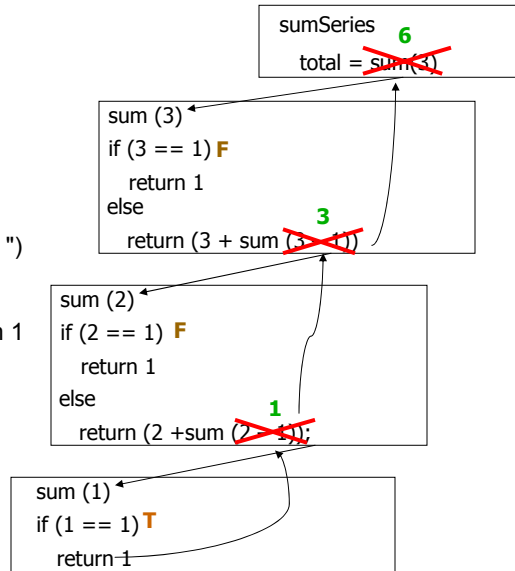
## **Requirements For Sensible Recursion**

- 1) Base case
- 2) Progress is made (towards the base case)

James Tam

## Example Program

```
def sum (no):  
    if (no == 1):  
        return 1  
    else:  
        return (no + sum(no-1) )  
  
def main ():  
    last = input ("Enter the last  
        number in the series: ")  
    last = (int) last  
    total = sum (last)  
    print ("The sum of the series from 1  
        to", last, "is", total)  
  
main ()
```



James Tam

## When To Use Recursion

- When a problem can be divided into steps.
- The result of one step can be used in a previous step.
- There is a scenario when you can stop sub-dividing the problem into steps (recursive calls) and return to previous steps.
- All of the results together solve the problem.

James Tam

## When To Consider Alternatives To Recursion

- When a loop will solve the problem just as well
- Types of recursion:
  - Tail recursion
    - A recursive call is the last statement in the recursive function.
    - This form of recursion can easily be replaced with a loop.
  - Non-tail recursion
    - A statement which is not a recursive call to the function comprises the last statement in the recursive function.
    - This form of recursion is very difficult (read: impossible) to replace with a loop.

James Tam

## Example: Tail Recursion

- Tail recursion: A recursive call is the last statement in the recursive function.
- Name of the example program: tail.py

```
def tail (no):  
    if (no <= 5):  
        print (no)  
        tail (no+1)
```

```
tail (5)
```

James Tam

## Example: Non-Tail Recursion

- Non-Tail recursion: A statement which is not a recursive call to the function comprises the last statement in the recursive function.
- Name of the example program: nonTail.py

```
def nonTail (no):  
    if (no < 5):  
        nonTail (no+1)  
    print (no)  
  
nonTail(1)
```

James Tam

## Drawbacks Of Recursion

Function calls can be costly

- Uses up memory
- Uses up time

James Tam

## **Benefits Of Using Recursion**

- Simpler solution that's more elegant (for some problems)
- Easier to visualize solutions (for some people and certain classes of problems – typically require either: non-tail recursion to be implemented or some form of “backtracking”)

James Tam

## **Common Pitfalls When Using Recursion**

- These three pitfalls can result in a runtime error
  - No base case
  - No progress towards the base case
  - Using up too many resources (e.g., variable declarations) for each function call

James Tam

## No Base Case

```
def sum (no):  
    return (no + sum (no - 1))
```

James Tam

## No Base Case

```
def sum (no):  
    return (no + sum (no - 1))
```

When does it stop???

James Tam

## No Progress Towards The Base Case

```
def sum (no):  
    if (no == 1):  
        return 1  
    else:  
        return (no + sum (no))
```

James Tam

## No Progress Towards The Base Case

```
def sum (no):  
    if (no == 1):  
        return 1  
    else:  
        return (no + sum (no))
```

The recursive case  
doesn't make any  
progress towards the  
base (stopping) case

James Tam



## Using Up Too Many Resources

- Name of the example program: recursion2.py

```
def fun (no):  
    print (no)  
    aList = []  
    for i in range (0, 10000000, 1):  
        aList.append("")  
    no = no + 1  
    fun (no)
```

```
fun(1)
```

James Tam

## Undergraduate Definition Of Recursion

Word: re·cur·sion

Pronunciation: ri-'k&r-zh&n

Definition: See recursion

Wav file from "The Simpsons"

James Tam

## **You Should Now Know**

- What is a recursive computer program
- How to write and trace simple recursive programs
- What are the requirements for recursion/What are the common pitfalls of recursion