

COMP 10
EXPLORING COMPUTER SCIENCE

Lecture 8
Recursion

TODAY'S OUTLINE

Recursion

Revisit search and sorting using recursion

- Binary search
- Merge sort



WHAT DOES THIS CODE DO?

```
int message () {  
    int x = 1;  
    cout << "we are in message: " << x << "!" << endl;  
    message();  
    return (x);  
}
```

A function is *recursive* if it calls itself



RECURSION

Recall the steps for executing a function call in C++:

- Allocate space for called function's parameters and local variables
- Initialize parameters
- Begin function execution

Recursive function calls work exactly like regular function calls

- New set of parameters and local variables for each call

EXAMPLE: FACTORIAL FUNCTION

The factorial of a non-negative integer n , denoted by $n!$, is the product of all positive integers less than or equal to n

$$n! = n * (n-1) * \dots * 2 * 1$$

Note: $0! = 1$

Factorial is an example of a mathematical function that can be defined recursively

$$\text{Note: } (n-1)! = (n-1) * \dots * 2 * 1$$

$$n! = n * (n-1)!$$

RECURSIVE FACTORIAL IMPLEMENTATION

```
// Compute n!  
// i.e. the product of the first n integers  
  
int factorial(int n){  
    int result;  
  
    if (n <= 1){  
        result = 1;  
    }  
    else{  
        result = n * factorial(n - 1);  
    }  
  
    return result;  
}
```

RECURSIVE FACTORIAL IMPLEMENTATION

```
// Compute n!  
// i.e. the product of the first n integers  
  
int factorial(int n){  
    int result;  
  
    if (n <= 1){  
        result = 1;  
    }  
    else{  
        result = n * factorial(n - 1);  
    }  
  
    return result;  
}
```

Trace:

factorial(4)

4 * factorial(3)

3 * factorial(2)

2 * factorial(1)

1

2 * 1

3 * 2

4 * 6

24

Google

recursion

Search Advanced Search

Web Show options... Results 1 - 10 of about 3,070,000 for recursion [definition] (0.14 seconds)

Did you mean: recursion

[Recursion - Wikipedia, the free encyclopedia](#)

A visual form of recursion known as the Drosie effect. The woman in this image is holding an object which contains a smaller image of her holding the same ...
en.wikipedia.org/wiki/Recursion - Cached - Similar - [?] [x]

[Recursion \(computer science\) - Wikipedia, the free encyclopedia](#)

Recursion in computer science is a way of thinking about and solving many types of problems. In fact, recursion is one of the central ideas of computer ...
[en.wikipedia.org/wiki/Recursion_\(computer_science\)](http://en.wikipedia.org/wiki/Recursion_(computer_science)) - Cached - Similar - [?] [x]

Show more results from en.wikipedia.org

[Recursion -- from Wolfram MathWorld](#)

A recursive process is one in which objects are defined in terms of other objects of the same type. Using some sort of recurrence relation, the entire class ...
mathworld.wolfram.com/Recursion.html - Cached - Similar - [?] [x]

[recursion](#)

Definition of recursion, possibly with links to more information and implementations.
www.fll.nist.gov/div897/isg/dads/HTML/recursion.html - Cached - Similar - [?] [x]

[Mastering recursive programming](#)

Jun 16, 2005 ... Recursion is a tool not often used by imperative language developers, ... He introduces the concept of recursion and tackle recursive ...

WHAT MAKES RECURSION WORK?

A recursive definition has two parts

One or more recursive cases where the function calls itself

One or more base cases that return a result without a recursive call

Rules:

There must be at least one base case

Every recursive case must make progress towards a base case

SEARCH REVISITED

Is there a more efficient way than linear search to search for target x in the sorted array?

Search(A, 8, 45)

Initial array

| | | | | | | | |
|---|----|----|---|-----|----|-----|----|
| 3 | 12 | -5 | 6 | 142 | 21 | -17 | 45 |
|---|----|----|---|-----|----|-----|----|

Sorted array

| | | | | | | | |
|-----|----|---|---|----|----|----|-----|
| -17 | -5 | 3 | 6 | 12 | 21 | 45 | 142 |
|-----|----|---|---|----|----|----|-----|

RECURSIVE BINARY SEARCH

Key Idea:

Do a little bit of work, and make recursive call to do the rest

Details:

Look at midpoint, and decide which half of the range is of interest

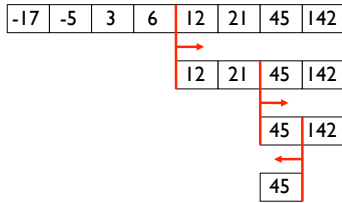
Use binary search again (recursively) to find value in reduced range

We can also think of this as a divide-and-conquer strategy

(...but not all recursion is based on divide-and-conquer)

EXAMPLE

Search(A, 8, 45)



BINARY SEARCH CODE

```
int BinarySearch(int A[ ], int first, int last, int target){
    int middle;
    if (first > last) return -1;
    middle = (first + last) / 2;
    if (A[middle] == target) return middle;
    if (target > A[middle])
        return binarySearch(A, middle+1, last, target);
    else
        return binarySearch(A, first, middle-1, target);
}
```

Where are the base cases?

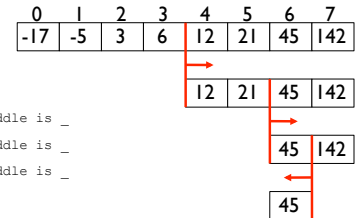
How does the call make progress?

ADD DEBUG STATEMENTS

```
int BinarySearch(int A[ ], int first, int last, int target){
    int middle;
    middle = (first + last) / 2;
    cout << "Searching A[ " << first << " .. "
          << last << " ]" << ". Middle is " << middle << endl;
    if (first > last) {
        cout << "First is larger than Last -- item not found\n";
        return -1;
    }
    if (A[middle] == target) {
        cout << "Middle " << middle << ". " << endl;
        return middle;
    }
    if (target > A[middle])
        return binarySearch(A, middle+1, last, target);
    else
        return binarySearch(A, first, middle-1, target);
}
```

TRACE EXECUTION

BinarySearch(A, 8, 45)



Searching A[_ .. _]. Middle is _
Searching A[_ .. _]. Middle is _
Searching A[_ .. _]. Middle is _
Middle _
Found 45 at 6.

What is the maximum number of calls possible?

SORTING REVISITED

Can we sort more efficiently than bubble sort?

Merge Sort is a recursive sorting procedure that employs a divide-and-conquer technique

Base case(s)?

How does the call make progress? What gets returned at the end of each call?

MERGE SORT

To sort an array of n elements:

If $n < 2$ then the array is already sorted,
Stop now.

Otherwise,

Sort the left half of the array.

Sort the right half of the array.

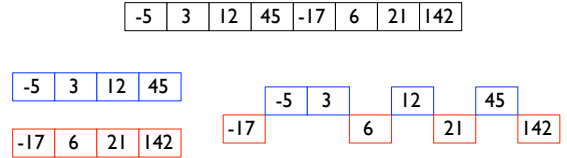
Merge the now-sorted left and right halves.

MERGE SORT ALGORITHM

```
void merge_sort (int A[ ], int low, int hi)
{
    // base case
    IF your array has only one element
        RETURN
    // recursive case
    ELSE
        // Sort the 2 halves of the array as follows:
        SET mid = (lo + hi) / 2
        merge_sort ( A, lo, mid )
        merge_sort ( A, mid + 1, hi )
        // combine the two halves
        merge ( A, lo, mid, mid + 1, hi )
}
```

MERGING TWO SORTED ARRAYS

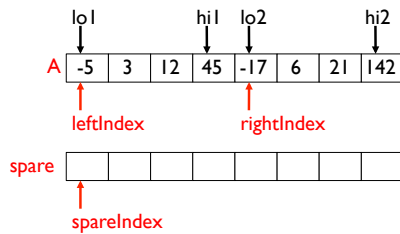
Efficient procedure



OUR MERGE PROBLEM

Need temporary storage

Need 3 indices to point to correct locations in arrays



MERGE ALGORITHM

```
void merge(int A[ ], int low1, int hi1, int low2, int hi2, int spare[ ])
{
    initialize the three indices

    // Initially need to grab numbers from both arrays
    WHILE there are still more elements to look at in both halves of the array:
        update the spare array with the smallest element from either half of the array
        update indices

    // At this point, all of the elements in one half of the array
    // have been copied into spare.
    IF there are more elements in the lower half of A:
        copy the elements in the lower half of A into spare and update indices
    ELSE IF there are more elements in the upper half of A:
        copy the elements in the upper half of A into spare and update indices

    // Now, all of the elements of A are in sorted order in array spare.
    Copy the elements of spare into A.
}
```

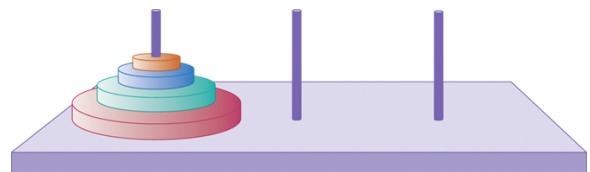
EFFICIENCY OF SORTING ALGORITHMS

Good algorithms are better than supercomputers

| | Insertion | Selection | Bubble | Shell | Merge | Heap | Quick | Quick3 |
|---------------|-----------|-----------|--------|-------|-------|------|-------|--------|
| Random | | | | | | | | |
| Nearly Sorted | | | | | | | | |
| Reversed | | | | | | | | |
| Few Unique | | | | | | | | |

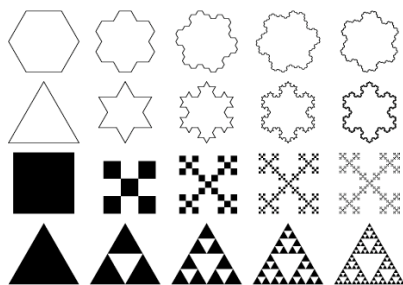
See sorting-algorithms.com

TOWERS OF HANOI



<http://www.cise.ufl.edu/~sahni/dsaaj/javaVersions/applications/TowersOfHanoi/TowersOfHanoi.htm>

FRACTALS



<http://mathworld.wolfram.com/Fractal.html>

How would you describe this fractal pattern?

How could you generate it?

```
*
**
*
***
**
*
**
*
*****
*
**
*
***
```

SUMMARY

A function is recursive if it calls itself

Popular recursive algorithms:

Binary search

Merge sort

