Reading: See Project 5 Handout: Finish Ch 2, Selections from 5, 6 and 10.

Maze (graph) search/traversal, and Queues in Discrete Event Simulation

as a views of the SAME IDEA

(needle-free language) Expressions

Recursion, Trees, Stacks, and

And half of this course covers:

CSI 310: Lecture 18
Mathematical Induction, careful expression of what programs should do, best "Reason" to solve problems.

(2) Techniques: data structures, nested documents/objects, and

(1) Understanding recursion: A function is recursive means the body of the function sometimes

Examples: fact(n), merge sort (char **, int n), your recursion...

We will review the 2 easy ways to make recursion: your friend: (1) "Faith"

(3) Removable Bl gaze.

(2) Techniques: data structures, nested documents/objects, and
CALLS.

ONLY think about the RESULTS of DIRECT RECURSIVE activities—DO NOT WORRY about activities during recursive.

(have faith that) run() will obey its POST-CONDITIONS. Then the "right thing" means after you verify run()'s PRE-CONDITIONS, the recursive call to run() does the "right thing" that every line-by-line with "faith" that every fact() etc.

Given a C++ coded function run(), e.g. merge_sort(), fact(), etc.:
thing on \( N \) elements to be sorted.

Simply put, \( N \) is a number for which \( \text{mergesort()} \) from DSQ does the right


Example of a property of a zero or positive integer \( N > 0 \):

\[
data[0..N-1] = [\text{Post-condition}]\]

\[
\text{Post-conditions Main and Savitch write for all cases of values in array}
\]

\[
\text{of DSQ's mergesort(). int *data, int N ) \text{ will satisfy the}
\]

that "faith" is based on the principle of mathematical induction, an axiom of

University at Albany Computer Science Dept.
This property is true for all counting numbers.

Then, people can conclude:

(2) If the property is true for all numbers less than N, then it is true for N itself.

Suppose it is verified for a property of counting numbers:

Then property is true for all counting numbers.

Principle of Mathematical Induction (CSI210 Preview):

University at Albany Computer Science Dept.
EACH valid input involves a particular NUMBER N. The „right thing“

matters: what valid inputs are given, will do „the right thing“.

RELY on Mathematical Induction to conclude that ALL the computations, no

critical; of course)
knowl.ode of C++ rules, careful reading of code, responsible memory use, etc is
deep thought; to verify that the code works.

USE the assumption („faith“) that the recursive calls WORK (do no

the function you are studying.

(1) Verify that the arguments to recursive calls involve FEWER elements than

(a) Verify that the code works in the „Base Cases“.

Application to study recursive programs:

University at Albany Computer Science Dept.
Don't forget this... 

\[ \frac{N}{2} + \frac{2}{N} = \frac{2}{1 + \frac{1}{N}} \]

...the fact that the graph of the function

\[ N(\frac{2}{1}) + \frac{2}{N} = \frac{2}{1 + \frac{1}{N}} \]

is an up ward opening parabola.

The sum of the first \( N \) numbers 1, 2, \ldots, \( N \) is true:

\[ \text{PROPERTY OF } N: \text{ The following equation is true:} \]

Typical application of Math Induction in Math courses like CS1210.
Combining is called "Merging".

1. Combine the two sorted groups into one large sorted list, and return it. This

**INDEPENDENTLY**

2. Divide the elements to be sorted into two groups of equal (or almost equal

size.

3. Sort each of these smaller groups (by recursive calls). That means

paradigm. The pattern is applied to the problem of sorting a sequence.

MergeSort is the algorithm that is invented when the divide-and-conquer
An list of I element IS SORTED (trivially)

{ if (istNodeSorted->next == NULL) return istNodeSorted;

    int n, n2; //to hold I ist and sublist lengths, maybe

    ListNode *answer2;
    ListNode *answer1;
    ListNode *subproblem2;
    ListNode *subproblem1;

    and local extent variables

    Value parameters *list2sort *list2sort *list2sort

    In your project's code, the automatic variables were:

    is done.

    keep track of what a particular activation of the (recursive) function Mergesort

    VERY IMPORTANT: LOCAL EXTANT (AUTOMATIC) variables are used to

    Spring 2007 Mergesort Project: Linked List variant.

    University at Albany Computer Science Dept.
return merge(ans1, ans2); Worked
ans2 = merge sort (subproblem2); Worked
ans1 = merge sort (subproblem1); Worked

\[ u > n \]
\[ u > n \]
\[ u > n \]
\[ u > n \]

Think: verify

\[ n = n - n \]
\[ n = n / 2 \]

\[ n > 1 \] case: somehow split the list. Perhaps find its length and calculate values
DIFFERENT ACTIVATIONS—DIFFERENT AUTOMATIC VARIABLES

The storage used for AUTOMATIC VARIABLES is in the activation record.

The EVENT of "calling one function once" is an ACTIVATION.

Review Way 2: Individual activations studied using a TREE DIAGRAM

Each time a function is called, an ACTIVATION RECORD is created.

Non-recursive function activations.

So, we do careful analysis of the SEPARATE INDIVIDUAL recursive or

University at Albany Computer Science Dept.
4. Now, automatic variables are in the activation record of the activation

RETURNED TO.

3. The \textsc{activation} record whose \textsc{call} originally created this activation is

\textsc{goes away}

\textsc{activation}'s activation record

2. \textsc{this activation}'s activation record

1. The return value (if any) is saved for use by the caller.

\texttt{return} \ldots\texttt{statement}:

\texttt{when a function activation

University at Albany Computer Science Dept.

12
Only true "logically"...this data is actually stored in the called activation.

ACTIVATION is destroyed, and its Record gets recycled.

Really: When an ACTIVATION executes the return; operation, that
spot within the function's body is this activation CALLED a function
Differently (one) whose CALL operation created this one. (3) The return
Definition of Activation Record: The data structure that holds (1) An
will control what THAT activation does.

Really: A new Function Activation is created, and the function's body
Wrong: Control "jumps" or "goes to" the function's body.

What HAPPENS when the computer executes a FUNCTION CALL?
sorts a list of 7 letters. 

Let’s review the operation of the top level activation of MergeSort when it
The two recursions have FINISHED; the top level activation will now merge.
Now, let's examine everything that happens during all the recursions.

Computation or activation was FINISHED.

Visual language convention: The indicated means "ghost": The indicated

Recurse.

SPLIT

MERGE

Recurse.

Recurse.

BROWN
To divide N/2 times you have to divide N by two with rounding to get

The number of levels is rounded \( \log_2(N) + 1 \).

The total number of elements involved in each LEVEL is \( N \) or fewer (the original input size).

This time COMPLETES the time used for recursions.

Operations is LESS THAN (constant) \( \times \) Number of elements.

During any one single activation, the time used to SPILT, MERGE and misc.

All the work is done by MERGE.

Why is MergeSort so fast?
When \( N = 2^{30} \) which is \( \log_2 N = 30 \), the total time is \( \text{LESS THAN} \)
particular activation of the (recursive) function `merge_sort` is doing.

Very important: automatic variables are used to keep track of what a
expression.

An expression tree illustrates the computations done when you evaluate an expression. The merge sort activation tree illustrated the computations done by merge sort.
\[(6+9)/3\times<6−4>\]
\[(6+9)/3\] \times \langle 4-6 \rangle \]
during the run of a recursive function.

during the run of a recursive function.

(Insert into a stack is called push. Delete from a stack is called pop.)

and non-recursive.

other data relevant to all C/C++ function calls and returns, both recursive

runs C/C++ programs: Implementing and organizing local variables and

2. Storing and organizing intermediate results when evaluating expressions.

parenthesized expressions.

1. Parsing which pairs of parentheses MATCH in a correctly nested

3 uses for stacks:

ONLY ONE END (called the top).

that access, insertion and deletion are permitted at

What is a stack? A stack is a sequence that is restricted so
A rooted tree is a structure of nodes and arcs (pairs of nodes) that has:

Zero or more rooted trees, with no nodes or arcs in common with each other.

One arc from this tree's root to the root of each of the trees specified under (a) One root node. (and)

One or more roots of trees. (b) Zero or more rooted trees, with no nodes or arcs in common with each other.

What is a tree?
An expression: 

(1) are substrings of the expression. 

(2) Any operator and operands under 

(3) and overlap(s) (and no 

(4) expressions as operands (no 

(5) if it has an operator, it has one 

(6) Binary (and 

(7) other or a top-level operator, 

(8) either is an identifier or constant 

A tree has 

under (q), the root of each of the trees specified 

(1) One arc from this tree's root to 

(2) other or the root. (and) 

(3) zero or more rooted trees, with 

(4) one root node. (and)
XXX employees.

smartness with the complex C/++ precedence/associativity rules, FIRE that

obvious. (2) If a programmer you are supervising times to show off his/her

2 Practical Rules: (1) If it's doubtful or subtle, USE PARENTHESES to make it

\[ \text{not}(3) \times \frac{5}{2} = \frac{7}{3} \times 4 \]

\[ 23 = 20 + 3 = \left( \frac{5}{4} \times 3 \right) + 3 \times 4 + 3 \]

precedence than addition. From elementary school:

must memorize or look up. They begin with "multiplication has higher

Rather complicated operator precedence and associativity rules people

How do you know which operator is evaluated first?

\[ A = B + C \times D + E \times F \]

Not fully parenthesized:

\[ A = B + C \times (D + E \times F) \]

Not fully parenthesized:

Expressions.

To make learning these ideas easier, we will start with fully parenthesized
The top level operation "Assign to A" is executed LAST! Why MUST it be done last? It uses the results of the all previous operations!

7. Assign it also to A.
6. Assign the last sum to B.
5. Add subtraction's result to this last product.
4. Multiply P and C.
3. Subtract that from C, remember result.
2. Multiply old value of E by D.
1. Increment E first.

means:

\[ A = (B = C) \times D - (E ++ \times P \times C) + (E ++ \times C) \]

Fully parenthesized:

\[ A = B = C \times D - E ++ \times P \times C \]
Example of an expression and its Parse Tree:

\[(A = (B = ((C - (D \times (E + F)) \times G)) + (F \times G)))\]
top level operator is multiplication

(\* \ r G)
(C-(D*(E++)))

- top level operator is subtraction

(D*(E++))

- top level operator is multiplication

(C-(D*(E++)))

- top level operator is subtraction

top level operator is increment

identifier

identifier

identifier
(p) An expression: A tree has:

- an identity or constant;
- or more operators or operands;
- or more expressions.

Either of these trees, with:

- zero or more rooted trees with
- other arcs in common with each
- tree root to this tree's root (and)
- one arc from this tree's root to
- the root of each of the trees specified (e)

(c) Any operator and operands under

- or more expressions (and) overlap(s).
An expression (p) are subexpressions of the expression.

(c) Any operator and operands under overlaps (and)

or more expressions (and) have one root node. It has one

slightly (and) or has a top level operator, except

Either is an identifier or constant.

A tree has:

definitions:

to be "trees and expressions" FIT these

Your job: Check that these examples, purporting
Expression det. Clause (p) is OK!

The operands are substituting:

Has an operator:  (C - (D \times (E++) ))

expression as operands, \( IS \) I or more which

\( \top \) level operator is multiplication ( \( * \) )

\( E \)

\( \top \) level operator is increment ( \( ++ \) )

\( (++) \)

\( (D \times (E++)) \)

is OK!

Clause (a)

is OK!

Clause (c) \( IS \) OK.

The operands are substituting:
and return its result.

(3) Combine the results from (2) using the meaning of the operator to compute

(2) RECURSIVELY Call Evaluate(L1) for each of the trees

_identifier. So, return it or its value.

(1) If L is just one node only, then the expression must be a constant or

Evaluate(ParseTree L)

tree:

The following recursive algorithm evaluates an expression when given its parse

The tree of an expression is called the expression’s Parse Tree.

The expression is called Parsing.

The (rather difficult and non-trivial) job of determining out the tree from a given

expression’s structure with

University at Albany Computer Science Dept.