Trees, Recursion, Expressions, Stacks

What are member functions, really?

The this pointer.

Deep versus shallow copy.

Accessing struct/class data members AKA fields.

Arrays, structures/classes, and pointers to structs.

CSI 310: Lecture 12
Selection using field name:

\[
\text{STT} \star \text{ps} = \text{new STT};
\]
Dynamic allocation by:

\[
\text{STT} \rightarrow \text{STT} \rightarrow \text{STT} \rightarrow \ldots \}
\]
Automatic storage allocation by:

\[
\{ \text{struct STT int addr; STT* link;} \}
\]
Struct definition:

\[
\text{struct selected by field (member) name addr}
\]
allocated near each other
nonuniform types defined by you
Collection of variables of
what's a structure (class type object) in
array type A?

\[
\text{assert (false \&\& A[2003]);}
\]
Dynamic allocation by:

\[
\text{etetype A[2003];}
\]
Automatic storage allocation by:

\[
\text{selected by subscript value in}
\]
allocated continuously
uniform type
Collection of variables of
what's an array A?
struct STT { int ddF; STT* link; ... }; STT G; STT* pS = new STT; STT* pA = new eleType[3];

pA = new eleType[3];
eleType* pA;

G

pS

pA

STT G; STT* ps; STT* link; STT* link; STT* link; STT* link;

struct STT { int ddF; } A[5];
```cpp
// printed 'Sappy
!

}()

#include <cstring>
#include <iostream>
```
sombody codes assigments and call-by-value.

customctor so YO programmers can program deep copying to happen when
Main and Savitch demonstrate overloaded operator().

Disadvantage of deep copies: the copy uses memory additional space.
Pitfall of shallow copies: Modifying the "copy" changes the original.

or fields.

A deep copy is made by allocating a new object for EACH object pointed to.

pointer variables point to the SAME OBJECT.

A shallow copy is made by just copying the pointer values, so for each, TWO
pointer fields.

Suppose you have a pointer variable or a structure containing one or more
Calling the "Fake Function member:"
{
    cout>>cout; //ERROR: dcl not in scope
    //prints the same thing!
    cout>>pf-dcf;
    cout>>*(pf-dcf);
}

void stl::smaller
    (stl* ps)
    
    // called, also it is in the class's scope and can access private members.

    // holding the address of the object for which it was called
    // which you can access via the keyword this

    // has an extra implicit

    // What does calling a function member like ps really mean? 

    ... stl C's stl* ps = new stl;
    stl ps = new stl;
    
    Suppose struct stl { void foo ( ) } ;

    

University at Albany Computer Science Dept.
by setting a breakpoint on it (you must \texttt{break main} or \texttt{quit} first).

1. If you step into a \texttt{MESS} (library function, etc.), you can pull the function was called.

\textbf{ACTION: WHEN DEBUGGING INSIDE A C++ FUNCTION}

\texttt{Command} \texttt{Step} \texttt{Steps} into functions when they are called, too.

\texttt{Function: Function calls are performed and return without stopping.}

\texttt{add command next} performs line-by-line controlled execution of the current

"this" is handy for debugging.
Recommended for beginners' (i.e., this course).

I. Detection of self-assigned or proper implementation of overloaded assignment operators. See page 174 of DS0 for details.

Some uses of C++ "this" pointer value in programming:

3. Defining of ostream::operator<< (\cdot\cdot\cdot) I/O member functions of istream::operator>> (\cdot\cdot\cdot)

4. Overloading prefix ++ operator with a member function (\cdot\cdot\cdot) returns X.

4. Application of programs.

Cin, etc. (they were called for. (Coded by C++ Library designers, not ct\n, etc.) These member functions return a reference to the istream object (cont.
instead of just friends of data type to output or input.):
But, running \texttt{fact(0)} calculates 0! = 0 is true and returns 1.

720! then computes 720! = 5040 and returns it.

For example, running \texttt{fact(7)} computes 7! = 5040, which returns

\begin{verbatim}

if (n==0) return 1;
else return fact(n-1)*n;
\end{verbatim}

Example: When the function runs, calls the same function, either directly or indirectly.

\textbf{Definition:} A function is \textbf{recursive} means the body of the function sometimes, re-implemented function itself.

(3) Elegant way to write programs, performance can be improved with runtime.

(2) Powerful problem solving technique.

(1) Understanding, not just programming, data structures and algorithms.

Recursion:

Trees, Recursion, Expressions, Stacks are closely related.

University at Albany Computer Science Dept.
One arc from the root to the root of each of the trees specified under or the root. (and)

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

(a) One root node. (and)

A rooted tree is a structure of nodes and arcs (pairs of nodes) that has:

What is a tree?
(b) are substrings of the expression. Any operator and operands under operator overlap, or more expressions are operands. If it has an operator, it has one level operator, exclusive. Either is an identity or constant. Either has more than one root. The root of each of the trees specified is zero or more rooted trees, with no nodes or arcs in common with each other. One root node. (and)
XXX employees.

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

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

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

\[ \text{not } (3 \times 4) + 3 \times 2 = 0 \]
\[ = 20 + 3 \times 4 + 3 = (3 \times 4) + 3 \]

mean

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?

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

Not fully parenthesized:

\[ \text{expressions.} \]

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 result of the all previous operations!

1. Assign it also to A.
2. Assign the last sum to B.
3. Subtract that from C, remember result.
4. Multiply F and G.
5. Add subtraction's result to this last product.
6. Assign the last sum to B.
7. Assign it also to A.

means:

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

fully parenthesized:

\[ A = B = C - D \times (E + F) \times G \]
Example of an expression and its Parse Tree
top level operator is multiplication ($F \ast G$)

identifier $F$

identifier $G$
The diagram illustrates the evaluation of the expression 

\[(C - (D \times (E++) +))\]

with the following top level operators:
- Subtraction (-
- Multiplication (*)
- Increment (++)

The diagram shows how the expression is broken down and evaluated step by step, starting from the innermost parentheses and moving outward.
(p) are subtrees of the expression.

(c) Any operator and operands under
overlapped.

(4) and no more expressions as operands (no
operator, it has one

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

(5) every (and) operator a top level operator, except-

(7) Either is an identity or constant.

An expression

(8) under (p).

The root of each of the trees specified

(c) One arc from this tree's root to
other or the root (and)

(4) no nodes or arcs in common with each

(6) zero or more rooted trees, with

(5) one root node (and)

A tree has:

definitions:

"trees and expressions" FIT these

Your job: Check that these examples purporting

University at Albany Computer Science Dept.
Tree det. clause (c) is OK!
other trees, nothing in common.

top level operator is multiplication (      )

identifier
d
d
d
identifier
identifier
identifier
c

The root node:

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

top level operator is subtraction (      )

identifier
d
d
d
identifier
identifier
identifier
c

Clause (c) is OK!
The operands are substituting:

Expression det. Clause (p) is OK!

Clause (c) is OK!

Expression det. Clause (p) is OK!

not overlapping.
Expressions as operands,
IS I or more
which

Identifier

Identifier

Identifier

Identifier

( (D*(E++))

(++)

(++)

(++)

(++)

(has an operator:

(identifier)

clause (a)

(identifier)

(fragment of a)

(subtraction)

(identifier)

(fragment of a)

(top level operator is multiplication (++)

(top level operator is increment (+))

(top level operator is increment (+))

top level operator is multiplication (*)

Expression det.

Expression det.
and return its result.

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

of the operands. (Only one call for a unary operator.)

identifiers. So, return it or its value.

if \( T \) is just one node only, then the expression must be a constant or

The following recursive algorithm evaluates an expression when given its parse

Tree.

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

Parsing is called the expression's structure with

The tree of an expression represents the expression's structure with

absolute clarity.
Is it to **OBSERVE** the stack of **ACTIVATION RECORDS** during the run of a

(Insert into a stack is called **push**, delete from a stack is called **pop**.)

and non-recursive.

other data relevant to all C/C++ functions calls and returns, both recursive
runs C/C++ programs. Implementing and organizing local variables and

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

2. Storing and organizing intermediate results when evaluating expressions.

3. The "run-time stack" of activation records, internal to the system when it

recursive function.

Lab4 is to

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
```c
{ return 0;

    while (finished) { /* entire file was read. */
        cout << A >> endl;  /* a sorted string was printed. */
        Now, A[0...n-1] is sorted. */
    }

    Now, A[0] has the smallest char from A[1...n-1].
    { /* [?] A[?], A[?] */
        ( [?] A[?] A[?])
        for (i = 0; i < n; i++)
            for (i = 0; i < n-1; i++)
                if (A[i] < A[i+1])
                    A[i], A[i+1] = A[i+1], A[i];

    while (cin.getline(A, size) != 0)
    { /* size, A[?] */
        cin.ignore(100, 't'); int nch, i; char A[size];

        using namespace std; main()

        #include <iostream>
        #include <cstring>
        #include <string>

        //Selection sort demo: processes chars within the array A.
```
For next week, after the midterm, we will implement the (recursive) MergeSort algorithm, a topic covered in the previous lectures.

The other half is to implement the SelectionSort algorithm. We will illustrate what half your Project 3 work must do.

C-strings

sort the names of landmarks in Project 2, "main list" lexicographically, as

Project 3 consists of implementing two sorting algorithms and applying them to

It uses the SelectionSort algorithm for sorting.

This program manipulates characters as if they were numbers!
Sample list of items to sort:

- Cat
- Aardvark
- Bat
- Ape
- Caterpillar
- Dog
- Ant
- Zebra
EXTRA STUFF...for review or future
or composed of its individual variables, taken together.

It is useful to consider the WHOLE ARRAY as ONE VARIABLE that is formed.

C/C++ the indexes range from 0 to Length-1.

• Each element is selected for access using an integer, called an index. In

  C/C++ the indexes range from 0 to Length-1.

  • Each element is selected for access using an integer, called an index. In

The elements are located contiguously in memory, at adjacent

• The number of elements (length of the array) is fixed.

  The elements are located contiguously in memory, at adjacent

  • The number of elements (length of the array) is fixed.

  char, any other type...

• Each individual variable, called an element, has the same type (int, float,

  • Each individual variable, called an element, has the same type (int, float,

  char, any other type...

An array is a sequence of variables (plural) that:

Arrays, again.
Mathematical vectors, e.g., coordinates of points in 3 dimensions, C++ code

Depending on how you tell WHERE THE END IS.

One way to store/process string data is to use a C++ array of char,

This is called string data.

See "RIGHT HERE" : R, I, C, H, T, etc.

Something NEW re. CSI201 (Non-numerical "text" data, such as the

What can you use arrays for?
C-strings are different from C++ strings you get from #include <string>

\[ W[0] \text{ with:} \]

The C-string "ABCD" (4 letters) is stored in a TENCHT 5 (five, not 4) char

Strings in char arrays terminated with \0 are called C-strings

The null char is coded \0 in C++.

The char "\0" is coded \A, called the "null char."

One way to tell where the end of a string is: Just after the last element used for
#include <iostream>

using namespace std;

int main()
{
    char MyChArray[12];
    cin.getLine(MyChArray, 12);

    cout << "Hello World" << endl;

    cout << "Required in CSt10: Declaring a variable that can hold a C-string:
C-strings are very easy to use. You have used them in CSt201 code like:
"

    cout << MyChArray << endl;

    return 0;
}
How can we very efficiently swap the strings in the nodes pointed to by I and J?
Swap the values in the 2 data fields of the nodes pointed to by I and J.

Computer DOES NOT copy chars nor node pointers!

char * pT;
pT = I->data; I->data = J->data;
J->data = pT;
and finally d. 

prints the characters 'in order, H, then e, two T's, an o, a space, then W, o, r, I,' 

count >> "HEllo World". 
is easier to think about than 

prints the string HEllo World 

count >> "HEllo World".

C-string, 

array of char (sequence of char variables) is a single variable that holds ONE

The examples of simple C-strings use illustrate the usefulness of thinking that an