Null-Terminated Strings, or "C-Strings" Sec. 4.5

Characters, char type and binary.

Getting started on Project 2.

CSI 310: Lecture 10 CSI 310: Showed out Lecture 8
That's the code for step (1)

```c
const int INBUF_SIZE = 72;

int main()
```

```c
{ cout << "ERR..." << endl; exit(0); };
if (cin.getline(input, INBUF_SIZE))
```

```c
....

char input[INBUF_SIZE];
```

```c
}
```

```c
using namespace std;

#include <cstring> //C-string library
#include <cassert>
#include <cstdio>
#include <cstdlib>
#include <iostream>
```
(6) After exhausting "known" commands, print a user error message and
is recognized, run code to do it and continue with step (1).
(5) If not, test if it's another implemented command, etc. Each time a command
(4) If so, exit.
          (                      
(3) If TRUE, we have a command. Is it: quit? Test with
          Test if input[0] == 'quit'?
(2) If TRUE, evidence to quit.
In place of the NEWLINE char, gettine() puts a \0?
array. All the characters the user typed up to THE NEWLINE are copied into the
file.
After a successful step (1), input is partially filled with the chars read in. The
p->data = new char [. . . [ strcpy(p->data, input) ; will do it!

returned by the above new operation.

What does your program actually store in the data field of a doubly-linked-list node for this project? The address of the C-string, i.e., the pointer value

node for this project? The address of the C-string, i.e., the pointer value

doesn't count the null char.

since "length of a string" doesn't count the null char.

Allocate dynamic storage for the new line:

iii(1) iii(1)

ADD ONE

III(1) III(1) String(input)

Use your own String function, or simply call

strlcpy(input)

How long a char array is needed to store the line?

(6) (7)

What if input[0] = "? Then input contains a line we must insert

into the main list. If so, do steps (6), (8), (7), and then continue with step (1)

(6)
If we began by designing the edtorcore class, that wouldn't have happened. We forget to INITIALIZE the data structure storing the main list.
Remember: A C-string is just an ARRAY of char values.

( )

The command language was designed so this place is easy to

(2) STAM a \0, in place of the char JUST AFTER the argument, to make it

where I is the position where the argument starts.

char *pattern = &input[I];

Answer: (1) Obtain the address of the first char of the string with:

strstr() which tests if a target string contains a pattern string.

How do you pass a string argument you found to various C-string functions, like

commands:

array? (Of course, run this algorithm after checking for all the “hexed”

Answer: YOU design an algorithm to scan those chars as they sit in the input

TIME number<Go from a prompt to a file>

How do you detect them? (Specifically, tell a pattern string<Go from a

certain commands (16-19) are not “hexed”; they have arguments.

University at Albany Computer Science Dept.
6. All data in a program's process is stored and computed in (current)

What printers print int conversion value

give the correspondence

7-bit ASCII character set given in Appendix A of DSO. ThisASCII table

of the most popular characters, about which most the World agrees, are the

C/C++ implementation.

Depending on VIEW

8-bit chars convert to 0 to 255, or to -128 to 127, depending on VIEW

4. chars are automatiically converted to and from ints. The 256 = 2^8 different

3. A char has at least 8 bits, guaranteed. 8-bit chars are almost universal.

So, by definition, sizeof (char) = 1.

2. Sizes of C/C++ variables are expressed as multiples of the size of a char:

`char` is a character literal.

1. A variable of type char can hold a character of the implementation character

Structure: C++ is character.

Programming Language

University at Albany Computer Science Dept.

Character set and type char
E is 69

... count >> ch >> " is " >> I >> end;

int I = ch; // char to int conversion.
char ch = 'E';

... So count means std::cout
using namespace std; // Get definition of std::cout
#include <iostream>

Why Letter E?

ASCII printer

make it add

make it print

sent to an

sent to an

computers in terms of binary digits or bits.

These same

These same

Base 2

Base 2

Why 69?

Why 69?

numeral

numeral

system

system

1 + 4 + 64

1 + 4 + 64

8 bits

8 bits

unit

unit

adding

adding

arithmetic

arithmetic

Why 69?

Why 69?

make it add

make it add

E

E

ASCII printer

ASCI printer

make it print sent to an

make it print sent to an

Social convention

Social convention

manufacturers;

manufacturers;
of the printer

of the printer

of the printer

of the printer

Base 2

Base 2

Base 2

Base 2

Why 69?

Why 69?

Why 69?
```c
{
    /*TRUE*/
    return N;
    
    {
        N++;
        /*TRUE*/
    }

    /*TRUE*/
    )

    while (p[N]) // equality with p(N) != 0
    
    }

    size_t strLen(const char *p)

    C-strings are often accessed through char * type vars.

    Unix and other system interface libraries use C-strings.

    #include <cstring>
    library has very useful functions.

    #include <iostream>
    facilities „know about“ C-strings.

    A C-string is a null-terminated array of char.
```c
{ 
    return target;
    The last copied char was \0
    
    { 
        TRUE \again.
        I++;
    }
}

if (...) {
    // Whoops, it is FALSE now!!!
    then test
}

while (0 == (target[I] = src[I])) {
    // tricky: copy first, then test
    I==#chars copied so far.
    size-I=0;
}

char * strcpy(char target[], const char src[]) 
```
target[I]=src[I];

If target[I] == 0?  is

YES

I+=1

NO

while(0!=(target[I]=src[I]))

I++;

The control-expression of the while-statement

The body of the while-statement

target[I]=src[I];

This character operation is always done at least ONCE!

It's important to understand "while" precisely...

strcpy(char target[], const char src[])

This character copy

The control-expression of the while-statement

return target;
INTERNET WORM DID THIS on purpose!

If the target array is non-trivial, the function return might be corrupted. THE FIRST
record stack, and so function return might be corrupted. THE FIRST

those legally occupied by the target array.

MEMORY located at addresses larger than (i.e., after)

chars at or after src, strcpy overwrites the

when the target array is SMALLER than 1+null

DANGER!!

University at Albany Computer Science Dept.
29 ordinary characters plus the 1 null terminating character!

```
assert (strlen(ACstr) == 30)
```

```c
char ACstr[]="I'm a string of 29 characters\n\0";
```

29 ordinary characters plus the 1 null terminating character!!

```
assert(sizeof(ACstr)==30)
```

Inside a block... ACstr is an automatic variable.
assert(sizeof(pACstr)==4) on many systems. NOT 30.

On the prev. slide ACstr[] WAS DIFFERENT

char * pACstr = "I'm a string of 29 characters!"

pACstr is a pointer variable initially pointing to a "constant" array

pACstr is an automatic variable inside a block.
Backward Compatibility

```c
char [] ;
and convert it to const char making the VERY BAD lines illegal.

Why don't standard C and C++ make the type of "a String literal" be const

// others will crash.
// or not on different systems.

Hello greeting which be copied!

strcpy(greeting, "bye-bye");
strcpy("bye-bye", greeting);
strcpy(greeting, "Hello");
strcpy("Hello", greeting);
strcpy(greeting, "Hello");
// OK but Dangerous.
...;

cchar greeting[ENOUGH-SPACE];
	null-terminated.

compiler-generated static "pseudo-constant" array filled with the given chars and

a string literal IS CONVERTED TO THE ADDRESS OF a
or sizeof("a String literal") or sizeof("A")

except when used like char array[ ] = "a String literal";

// moving C++ thing...
```

University at Albany Computer Science Dept.
What are member functions, really?
Pointers to structs and accessing data members AKA fields.

CSI 310: Lecture 10.5
<table>
<thead>
<tr>
<th>pps, dpp or ps-&gt;dpp</th>
</tr>
</thead>
<tbody>
<tr>
<td>.dpp</td>
</tr>
<tr>
<td>c.dpp</td>
</tr>
</tbody>
</table>

Selection using field name:

STT* ps = new STT;
Dynamic allocation by

STT* g; ...}
Automatic storage allocation by

{ struct STT int dpp; STT* tink; }

struct definition:

selected by field (member) name dpp
allocated near each other
nonuniform types defined by Y0W
Collection of variables of

What's a structure (class) type object (get) ?

[ ] p

assert (?);

[ ] a

assert (0<=j <=2003);
Selection using array notation:

etertype *pa = new etertype[
  etertype a[2003]; ...
];
Automatic storage allocation by

selected by subscript value
allocated contiguously
uniform type
Collection of variables of

What's an array A?
struct STT {
  int ddF;
  STT* link;
};

STT G;

STT* pS;
pS = new STT;

STT* STT* STS;
{ ... STT* link; ... };

struct STT* STS{
  int ddF;
};

eleType A[5];

pA = new eleType[3];

STT ps
ps = new STT;

STT* STT* STS;
{ ... STT* link; ... };

struct STT* STS{
  int ddF;
};

eleType A[5];

pA = new eleType[3];
Calling the "Fake function member":
{
    cout << "ERROR: ddf not in scope \n";
    //prints the same thing!
    cout >> p->ddf;
    cout >> this->ddf;
    cout >> (p) * ddf;
}

(void FAKEx(STT *p))
(void FAKEx(STT))

Calling the function member:
{
    //prints the same thing!
    cout >> this->ddf;
    cout >> this->ddf;
    cout >> ddf;
}

What does calling a function member like FUN really mean?

...; STT G; STT *ps = new STT;
{;
    int ddf;
    :: Fake (ps);
    :: FaKe (G);
    ...
Suppose struct STT {
    void FUN ;
}
by setting a breakpoint on it (you must `break main` or `catch` and `run` first). `break main` will print the name of the function in which the execution was halted. `print this` will print the object for which the activation was called. The tips:

- `print this` prints this.
- `function activation`:
  - when debugging inside a C++ function
  - command `step` steps into functions when they are called, too.
  - command `next` performs line-by-line controlled execution of the current function.
  - function calls are performed and return without stopping.
  - command `dd` command is handy for debugging.

Tips: 1) If you step into a `MESS` (library function, etc.), you can pull the output into a `MESS`...
4. Overloading prefx ++ operator with a member function (++) returns X.

4. Overloading prefx ++ operator with a member function (++) returns X.

Application programs:

- These member functions return a reference to the `iostream` object (cout, cin, etc.) if the `>>` operator is called for. (Coded by `operator>>` in the library, not `operator>>` in the library.

- These member functions return a reference to the `iostream` object (cout, cin, etc.) if the `>>` operator is called for. (Coded by `operator>>` in the library, not `operator>>` in the library.

- These member functions return a reference to the `iostream` object (cout, cin, etc.) if the `>>` operator is called for. (Coded by `operator>>` in the library, not `operator>>` in the library.

3. Defining `iostream` `operator>>` (i.e., this course).

Recommended for beginners: Advanced `operator>>` syntax:

Occasionally required for other objects.

- Some member functions will store the address of the current object into

- Some member functions will store the address of the current object into

I. Detection of self-assignment for proper implementation of overloadead

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

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

University at Albany Computer Science Dept.