Pre-reading function header

If, loops, functions

Basic C++ Syntax

Not T/F M/C
Closed notes
Closed book

Quiz

Midterm
FOUR OPEN ("inv.dat");

FOUR CLOSE

OUTPUT FILE

IF STREAM FILE

WRITE "PUT INTO ARRAY OF FACTORS"

IF N = 1

WRITE "PRINT FACTORS"

WRITE "CAI N ENTER"

READ Cin;

EXIT PROGRAM

IF STREAM CAI;

IF STREAM GET;

EXIT PROGRAM
2. char frame[213];

foot. open ("frame")?

street (frame, "data")?

cin >> frame (cin. ignore(c));

foot. open ("inv.dat")?

ostream foot;

a file: #include <fstream>

To write out file ... to write to
if foot_close() > foot_close_clear
{
    foot << variable << 'n';
}

if (foot) // was open successfully
    foot is False (zero) otherwise
    open vowel successfully.
    // foot is True (non-zero)!

foot open (if variable, is: app)

To keep the flu in fact:
3.
Input from a file

---

```
while (fin != fin.eof())
{
    line = fgets(line, 80, fin);
    // a file to read from
    if (line == NULL) // most likely this is
        // fin.open("filename")
    else
        printf("%s", line);
}
```

---

Character string of something
False - otherwise

Operation failed

True - if the previous input

Fin. etc.
\[ \text{int } \\
\text{ptr} \]

\[ \text{ptr} = \text{new int} \]

1. \text{ptr} is a pointer to an int.

2. int *ptr;
cin >> name;

char* array = new char[7];
array = dynamicaly

Array

Stringly Allocated
8

"complete time"

cout << name << endl;

Arrays
int i = 0; 
while (i < length) { 
    if (ptr[i] == NULL) 
        ptr[i] = new int; 
    ptr[i] += ptr[i] + ptr[i+1]; 
} 
free(ptr);
If `curr == prev`:

- If `prev == NULL`:
  - No
  - End

- No
  - Yes

Delete `prev`.

If `prev` is pointing at
release its memory.

End.
\text{Once}\ \text{done}\ \text{delete temp.}

\text{Once}\ \text{int}\ \text{arr}\ \text{allocated}\n
\text{temp = ptr.}

\text{ptr = new int;}

\text{int \* ptr, \* temp;}

1.
Dynamically Allocated Arrays
Address

2 * sizeof (char)

(name + 2)

(name[2] = ', p,

Continuous

name

```
... |
```

13
$\phi \cdot \phi = (**b** \cdot (**p** \cdot \text{char} \text{char} \text{char}))$

$\textbf{while} (**p** \cdot \text{char} \text{char} \text{char})$

\textbf{step} (**char \text{char} **p** \cdot \text{char} \text{char})