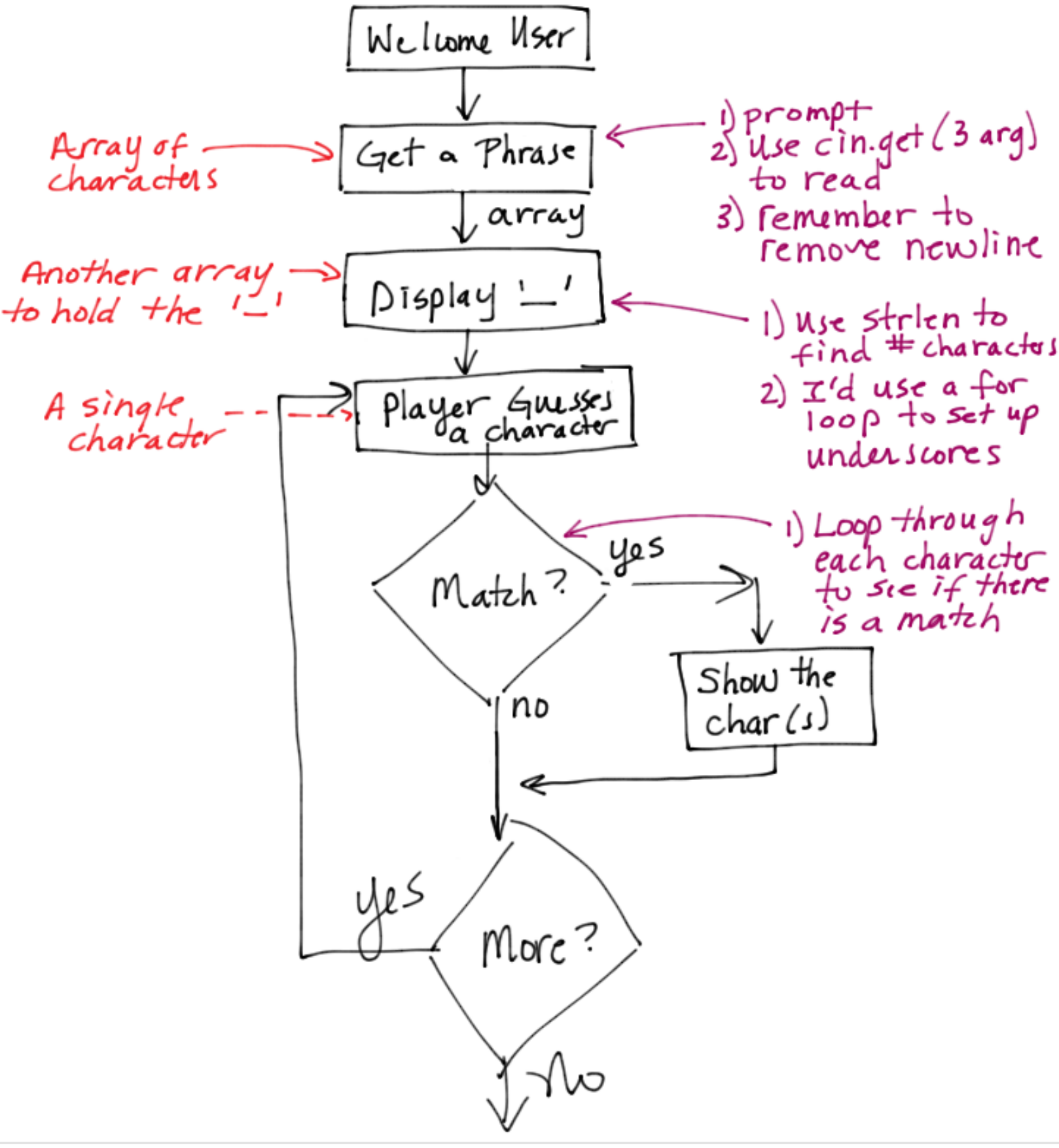
Today in CS162 - Lectures 3 & 4

1. Continue with Topic #1, if's, loops & arrays (page 11 of Day #2's handwritten slides)
2. Create, read, display arrays
3. Lab: create a hangman program

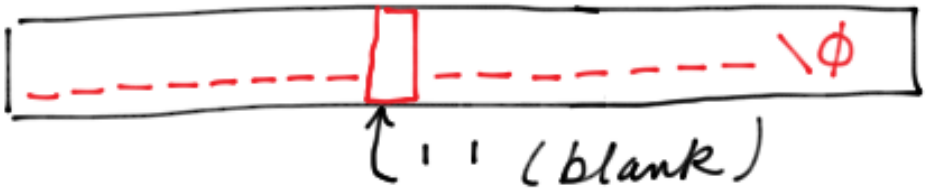
Announcements

Design a Hangman Program:



1) Computer Science \0 ← Original Array

2) int length = strlen(array); ← 16

3) 

← where the Resulting "Guess" is built

```
int i;
for( i=0 ; i<length ; ++i )
{
    if(array[i] != ' ')      // not a blank
        answer[i] = '_';
    else
        answer[i] = ' ';     // Keep the blank
}
answer[i] = '\0';
```

↑ notice to check your stopping conditions

← Important

if i had been defined in the initialization step of the IF then i would not be available

Preview of Functions

Prototype: Declares the function (so it can be called before the function exists)

- with a prototype we don't have to worry about the order in which we create our functions

return_type function_name (data_type arg_name);

- return_type: void when nothing is returned
- function_name: must be a valid identifier
- arg_name: optional
- arguments are a comma separated list

```
void calculate (int hours, float wage, char name[]);
```

- int hours, float wage: Pass by value
- char name[]: when passing an array

When passing an array, the address of the first element is actually passed BY VALUE (pass by value means a copy is made ...)

Function Calls:

1. () is the function call operator
2. Calculate(num, wage, name);

notice in a function call there are no data types or [] listed

3. If we are calling a function that returns a value, then you need to "catch" that value

```
char toupper(char);
```
← from the cctype library

Function call:

```
char variable = 'a';
variable = toupper(variable);
```

what would happen if we left this out?

Function Definitions

For the hangman game, we will write the following functions:

1. `void read_phrase(char phrase[]);`
   ← the return type cannot be an array

2. `void setup(char result[], char phrase[]);`
   '_'

3. `char guess();` ← returns the single character guess

4. `bool match(char guess, char result[], char phrase[]);`
   returns true if there is a match and replaces the character

5. `bool finished(char result[]);`
   returns true when there are no '_' in the resulting array.
   (— OR when strcmp of the two arrays is the same! In that case you would need to send in both arrays)

Next... create the code

1. Keep main small by using functions

2. Overall structure:

- read the phrase
- set up the resulting array
- do
  {
    output the '_' array (result)
    guess a character
    any matches, assign based on the original array
  } while (!finished(resulting array));