COMP1511 PROGRAMMING FUNDAMENTALS

LECTURE 14

Let's look at some linked list problems and how to solve them...

LAST TIME...

- Deleting from a linked list
  - nowhere to delete (empty list)
  - at the head
  - at the tail
  - in the middle!

TODAY...

- Recap deleting in detail
- Talk about leak checking
- Some linked list problems:
  - Delete at n'th position
  - Find and print out all even numbers in a list
  - Insert nodes in order in a list
- Talk about multi-file projects

"

Live lecture code can be found here:

HTTPS://CGI.CSE.UNSW.EDU.AU/~CS1511/22T3/LIVE/WEEK08/

LINKED LISTS

RECAP: DELETING A NODE

- In all instances, we follow a similar structure of what to do when deleting a node. Please draw a diagram for yourself to really understand what you are deleting and the logic of deleting in a particular way.
- To delete a node in a linked list:
  - Find the previous node to the one that is being deleted
  - Change the next of the previous node
  - Free the node that is to be deleted
  - Consider possible edge cases, deleting if there is nothing in the list, deleting when there is only one item in the list, deleting the head of the list, deleting the tail of the list, etc.

PROBLEM TIME

PRACTICING WITH LINKED LISTS

Problem 1: Delete at the nth position of a linked list.

The user will be given the position at which to delete and they will need to remove the node at that position from the specified linked list.

So for example, given the list: 3 -> 4 -> 5 -> 6 -> NULL and delete at position 3:

The list will become 3 -> 4 -> 6 -> NULL

PROBLEM TIME

PRACTICING WITH LINKED LISTS

Problem 2: Given a linked list of numbers, find all the even numbers in the list and print them out (extension if time: make a linked list of the even numbers from the list)

a) The user will be given a linked list of numbers and they need to find all the even numbers and then print those out:

So for example, given the list: 3 -> 4 -> 5 -> 6 -> NULL:

The output would be 4, 6

PROBLEM TIME

PRACTICING WITH LINKED LISTS

Problem 2: Given a linked list of numbers, find all the even numbers in the list and print them out (extension if time: make a linked list of the even numbers from the list)

b) The user will be given a linked list of numbers, and they will find all even numbers and make a linked list out of them.

So for example, given the list: 3 -> 4 -> 5 -> 6 -> NULL

The output would be 4 -> 6 -> NULL

BREAK TIME...

A prisoner is locked up in a tower with two doors. One of them leads to the exit, the other to the oubliettes. A guard is stationed in front of each door. One of them always tells the truth, the other lies systematically.

What one question must the prisoner ask just one of the guards to be certain of finding out which door leads to freedom?

PROBLEM TIME

PRACTICING WITH LINKED LISTS

Problem 3: Given a some numbers, insert them in ascending order into a linked list

So for example, given the list: 3 6 5 4

The output would be 3 -> 4 -> 5-> 6 -> NULL

MULTI FILE PROJECT

WHAT ARE THEY?

- Big programs are often spread out over multiple files. There are a number of benefits to this:
  - Improves readability (reduces length of program)
  - You can separate code by subject (modularity)
  - Modules can be written and tested separately
- So far we have already been using the multi-file capability. Every time we #include, we are actually borrowing code from other files
- We have been only including C standard libraries

MULTI FILE PROJECT

WHAT ARE THEY?

- You can also #include your own! (FUN!)
- This allows us to join projects together
- It also allows multiple people to work together on projects out in the real world
- We will also often produce code that we can then use again in other projects (that is all that the C standard libraries are - functions that are useful in multiple instances)

MULTI FILE PROJECT INCLUDES

.H FILE
.C FILE (MAYBE MULTIPLES)

- In a multi file project we might have:
- (multiple) header file - this is the .h file that you have been using from standard libraries already
- (multiple) implementation file - this is a .c file, it implements what is in the header file.
- Each header file that you write, will have its own implementation file
- a main.c file - this is the entry to our program, we try and have as little code here as possible

header
file

#include "   .h"

impleme
ntation
file

.c

HEADER FILE
#INCLUDE "SOMETHING.H"

Typically contains:

- function prototypes for the functions that will be implemented in the implementation file
- comments that describe how the functions will be used
- #defines
- the file basically SHOWS the programmer all they need to know to use the code
- NO RUNNING CODE
- This is like a definition file

IMPLEMENTATION FILE

SOMETHING.C

This is where you implement the functions that you have defined in your header file

IMPLEMENTATION FILE

MAIN.C

This is where you call functions from that may exist in other modules.

AN EXAMPLE

A MATHS

- We will have three files:
  - header file - maths.h
  - implementation file - maths.c
    - #include "maths.h"
  - main file - main.c
    - #include "maths.h"

maths.h
```
1 // This is the header file for the maths module example
2 // The header file will contain:
3 //      - any #define
4 //      - function prototypes and any comments
5
6 #define PI 3.14
7
8 //Function prototype for a function that calculates
9 //square of a number
10 int square(int number);
11
12 //Function prototype for a function that calculates
13 //sum of two numbers
14 int sum(int number1, int number2);
15
```
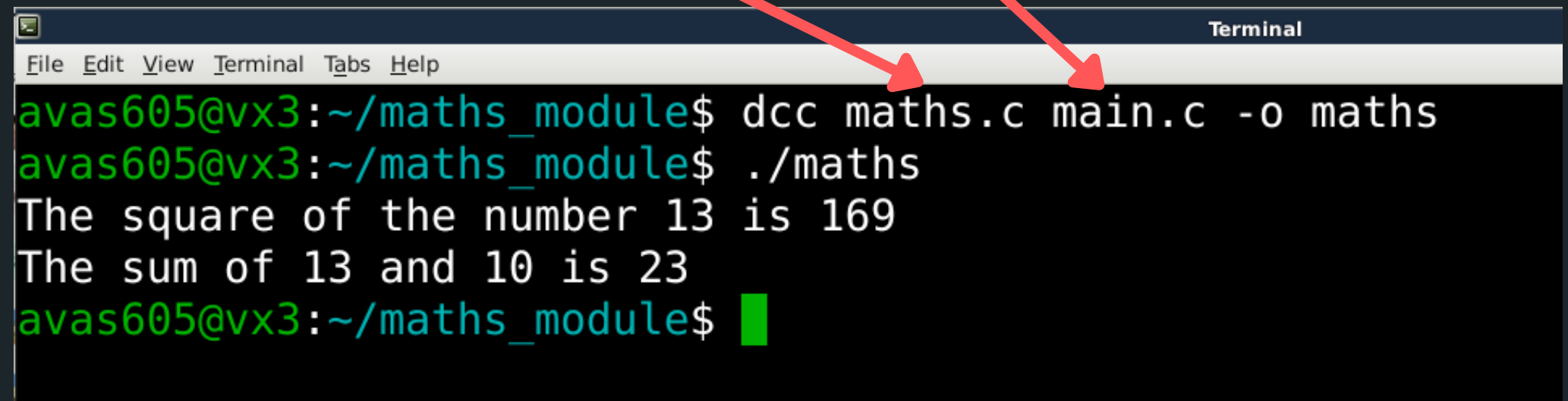
main.c
```
1 //This is the main file in our program
2 //This is where we drive the program from and where we
3 //make calls to our modules. We need to include the
4 //header file for each module that we want to use functions
5 //from
6
7 #include <stdio.h>
8 //Include the header file:
9 #include "maths.h"
10
11 int main (void) {
12     int number = 13;
13     int number2 = 10;
14
15     printf("The square of the number %d is %d\n", number, square
(number));
16     printf("The sum of %d and %d is %d\n", number, number2, sum
(number, number2));
17     return 0;
18 }
```

maths.c
```
1 //This is the implementation file of maths.h
2 //We defined two functions in the header file,
3 //and this is where we will implement these two
4 //functions
5
6 //Include your header file in the implementation file
7 //by using the below syntax
8
9 #include "maths.h"
10
11 int square(int number) {
12     return number * number;
13 }
14
15 int sum(int number1, int number2) {
16     return number1 + number2;
17 }
```

COMPILING A MULTI FILE

COMPILE ALL C FILES IN THE PROJECT

- To compile a multi file, you basically list any .c files you have in your project
    - In the case of our example, we have a maths.c and a main.c file):

- The program will always enter in main.c, so there should only be one main.c when compiling

Feedback please!

I value your feedback and use to pace the lectures and improve your overall learning experience. If you have any feedback from today's lecture, please follow the link below. Please remember to keep your feedback constructive, so I can action it and improve the learning experience.

https://www.menti.com/alpw3shkdjcx

WHAT DID WE LEARN TODAY?

LINKED LISTS

linked_list.c

main.c

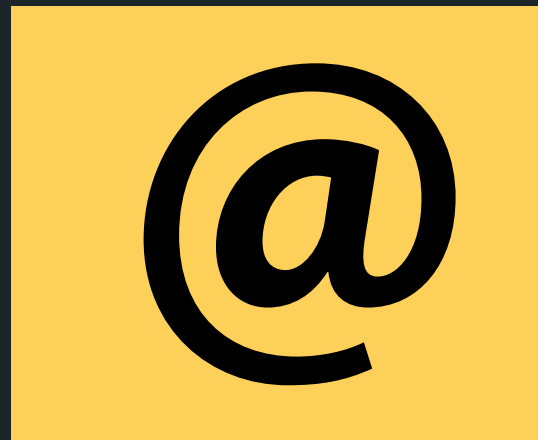linked_list.h

MULTI-FILE PROJECTS

maths.c

main.c

maths.h

REACH OUT

CONTENT RELATED QUESTIONS

Check out the forum

ADMIN QUESTIONS

cs1511@unsw.edu.au