COMP1511/1911 Programming Fundamentals

Week 10 Lecture 2

The Final Lecture

Assignment 2

We have decided to go ahead with the 1 day extension for assignment 2.

The deadline will now be **Saturday 16th at 5pm**.

We also have an FAQ on the forum
Assignment 2 FAQ Forum Post

My Experience Surveys

myExperience
Tell us about your experience and shape the future of education at UNSW.
Click the link in Moodle
UNSW SYDNEY

Please be mindful of the UNSW Student Code of Conduct as you provide feedback. At UNSW we aim to provide a respectful community and ask you to be careful to avoid any language that is sexist, racist or likely to be hurtful. You should feel confident that you can provide both positive and negative feedback but please be considerate in how you communicate.

http://myexperience.unsw.edu.au/

Week 10 Practice Exams

- If you are in an online tut-lab
    - you can sign up for an in-person lab
    - Access code: COMP1511
- It is run in a lab so goes for 2 hours
- Just submitting 1 question from lab or home is enough for lab marks
- Submissions from exam account won't show up on submissions web site

https://buytickets.at/comp1511unsw/1447098

The Exam: Time and Date and Location

- Date: 25th November
- 3 hours + 10 minutes reading time
- There will be 2 sessions of the exam.
- The students sitting the exam in the afternoon will be corralled for a period of twenty-thirty minutes as we conduct the changeover.
- Students in the morning exam, will not be able to leave early.
- **Mon 25/11/2024 Morning** (09:45-13:00)
- **Mon 25/11/2024 Afternoon** (12:50-16:40)

Supplementary Exam

The supplementary exam for the course is confirmed to be on

Tuesday Jan 21 2025
Exact times and locations will be organised closer to the date

Note: this is only for people who are granted special consideration due to illness or misadventure.

Week 11 Revision Sessions

The week 11 revision sessions will be run:

Monday 1pm - 3pm (f2f) in Piano (K14 Physics LG18)

Wednesday 12pm - 2pm (online)

Please sign up for the revision sessions (Access code: "COMP1511"):
https://buytickets.at/comp1511unsw/1414692

Study Resources

- Full Practice Exam Week 10 Labs
- Practice Exam questions Week 9 and 10 Lectures
  - Some we did/will do in lectures and you can try them again for yourself. See the _stub.c versions
  - Some have been left for self study
- Extra Challenges questions (some are 1 or 2 dots)
- Extra Revision questions
- Redo Problem Sets
- Try Lecture Coding problems yourself

Last Lecture

- Exam Revision
  - Strings, pointers and command line arguments
  - 1 and 2 dot hurdle questions
  - Look at the exam environment

Today's Lecture

- More difficult Linked List Questions
  - Debugging: search and delete approach 2
  - Split lists (Email Management System) (3 dot level)
- Sample exam questions
  - 2 dot Linked List and array hurdles
  - Debug string question
  - 3 dot array/string/command line argument question

There are many more questions than we will cover in the lecture.

These are for your own revision.

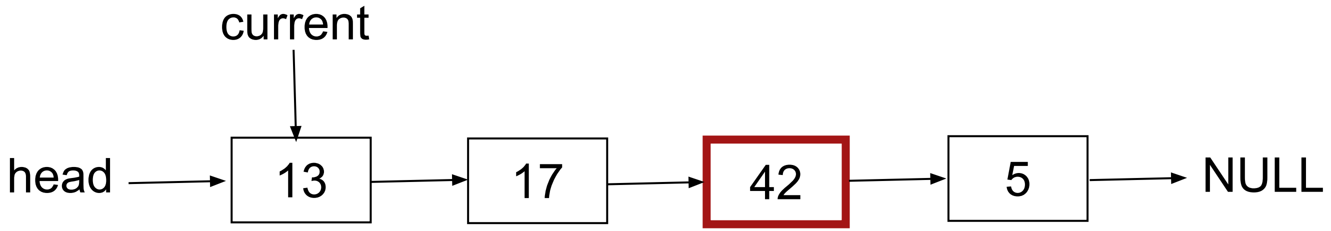
Linked List Coding Reminder

Some special boundary conditions that you need to consider when you manipulate lists:

- Empty list
- List with 1 element
- Something happening at the beginning of the list
- Something happening at the end of the list
- Something will not occur, the item is not in the list (inserting after a number that doesn't exist etc)

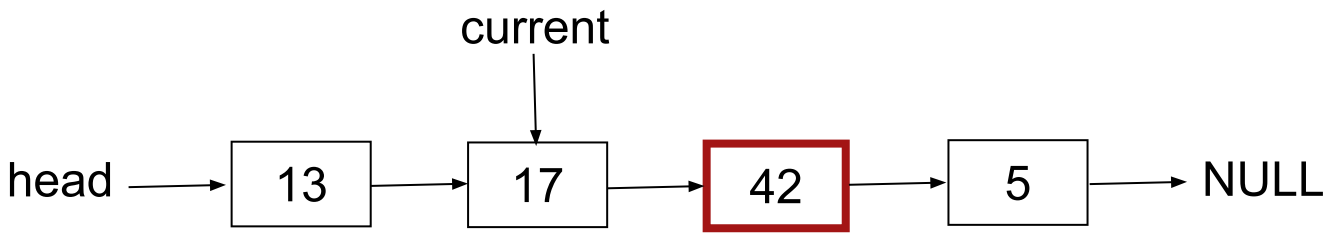
Search and delete Approach 2: general case

```c
// Approach 2: Just use 1 pointer to traverse
// but check the next node
struct node *current = head;
while (current->next != NULL &&
       current->next->data != search_key) {
    current = current->next;
}
```



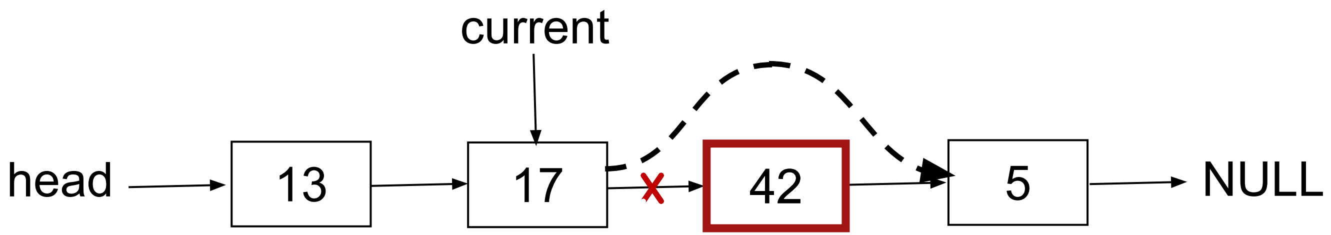
Search and delete Approach 2: general case

```c
// Approach 2: Just use 1 pointer to traverse
// but check the next node
struct node *current = head;
while (current->next != NULL &&
        current->next->data != search_key) {
    current = current->next;
}
```



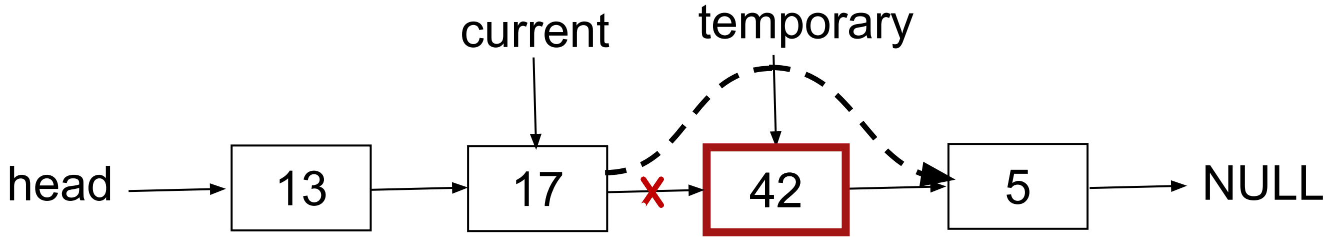
Search and delete: Approach 2

Then we need to connect current node to the one after the one we are deleting. But we still need a pointer to the node we want to free. How can we do that?



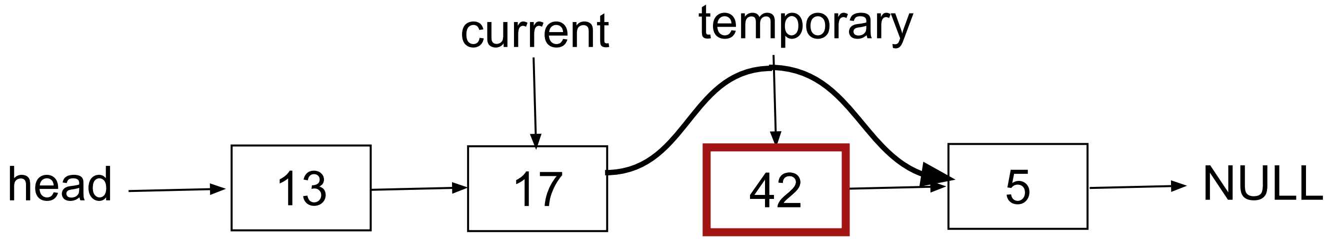
Search and delete: Approach 2

```
struct node *temporary = current->next;
```



Search and delete: Approach 2

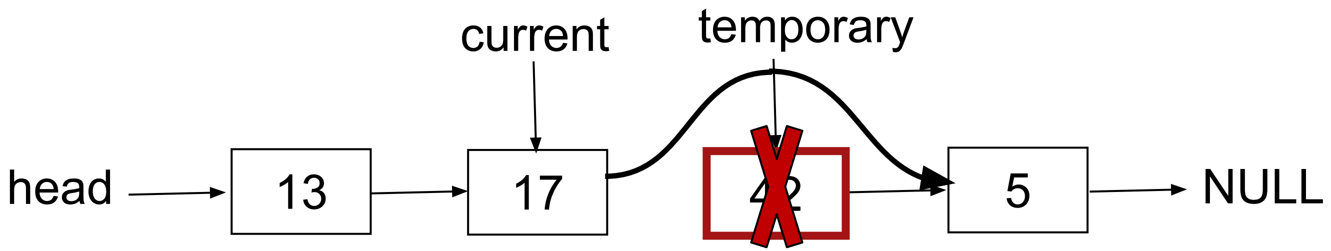
```
struct node *temporary = current->next;
current->next = temporary->next;
```



Search and delete: Approach 2

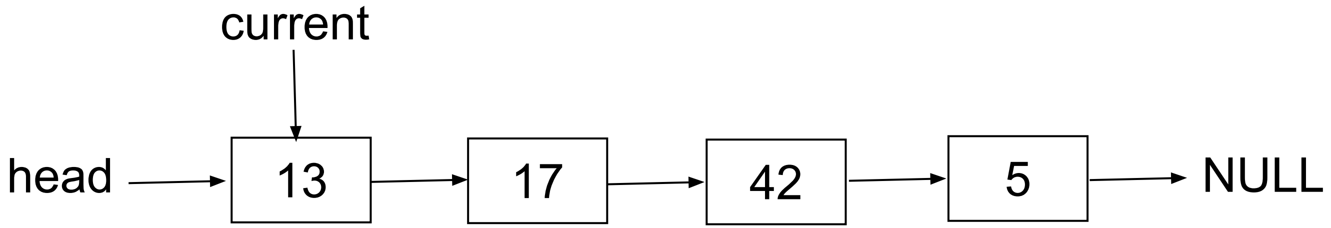
Now we can free the node we want to delete

```
free(temporary);
```



Search and delete Approach 2: Edge Cases

- Unfortunately we will need to debug our code as there are edge cases that have not been accounted for!
- It is always helpful to debug linked lists with a diagram to help visualise things



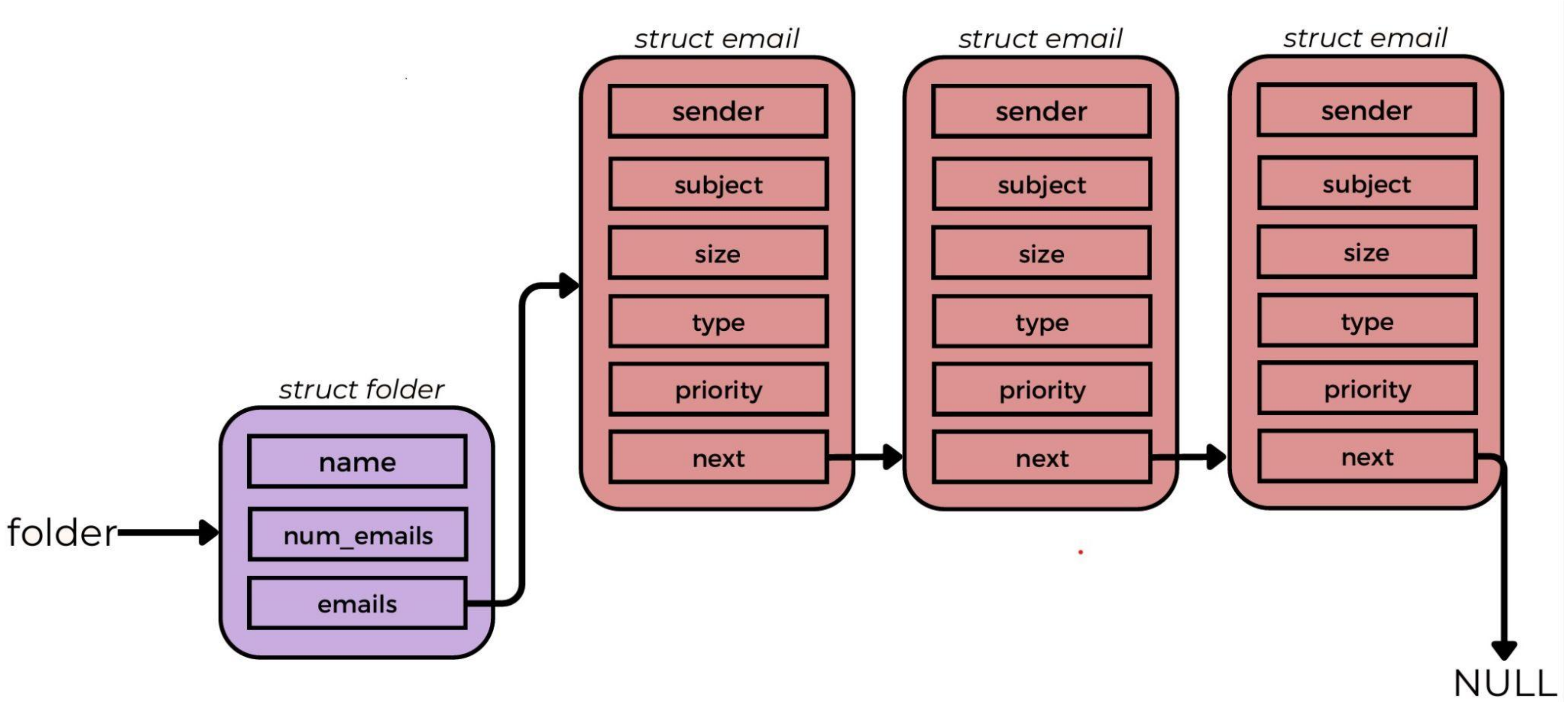
Email Management System (EMS): structs

```
struct folder {
    char name[MAX_LEN];
    //to use later :)
    //int num_emails;
    struct email *emails;
};
```

```
struct email {
    char sender[MAX_LEN];
    char subject[MAX_LEN];
    double size;
    enum email_type type;
    enum priority_type priority;
    struct email *next;
};
```

Exercise for you to try: Implement split_folder.c

EMS: Visualisation of the system



EMS: Linked List (●●●)

```c
// Split emails in a given folder into 3 folders based on
// email type keeping original ordering
// assumption: inbox, sent and draft are folders
// are already malloc-ed and initialised
void split_folder(struct folder *given_folder,
    struct folder *inbox, struct folder *sent,
    struct folder *draft) {
    // TODO: implement this function
    printf("split_folder not yet implemented.\n");
    exit(1);
}
```

Sample Hurdle Exam Questions

Linked List Hurdle Example (●◌◌)

`linked_list_range.c`

Find the range (the difference between the biggest term and the smallest term) of a linked list

This is an exercise for you to try

Linked List Hurdle Example (●◌◌)

`linked_list_equal.c`

Given two linked lists, return the number of values in the first linked list that are equal to the corresponding values in the second linked list.

Completed on Monday

Linked List Hurdle Example (●◌◌)

list_concatenate.c

Concatenate two linked lists (join one linked list to another)

New: This is an exercise for you to try

Linked List Hurdle Example (●◌◌)

list_find_divisible.c

Count all the elements in the linked list that are divisible by 6 and output the count.

New: This is an exercise for you to try

Linked List Hurdle Example (●◌◌)

list_even_difference.c

Given two linked lists, count the number of even numbers in both linked lists and return the difference.

New: This is an exercise for you to try

Array Hurdle Example (●◌◌)

`count_odd.c`

Given two arrays of the same size, return the number of items at corresponding indexes that are odd in both arrays.

This is an exercise for you to try

Array Hurdle Example (●◌◌)

`array_struct_capitals.c`

An array of structs of type:

```
struct initials {
    char first_initial;
    char last_initial;
};
```

return the count of all structs that have don't have capital letters for both of their initials.

Completed on Monday

Linked List Hurdle Example (●●◌)

`linked_list_duplicate.c`

Duplicate every node in the list by inserting the same node after the original node.

Completed on Monday

Linked List Hurdle Example (●●◌)

`linked_list_delete_duplicate.c`

Delete the first instance of a duplicate in the given list e.g.

1->3->5->3->1->7->X

would give

1->3->5->3->7->X

Linked List Hurdle Example (●●◌)

list_insert_ordered.c

Insert a new node into a sorted linked list, maintaining the sorted order.

New: This is an exercise for you to try

Linked List Hurdle Example (●●◌)

linked_list_delete_div_six.c

Delete the first node in the list that is divisible by 6

New: This is an exercise for you to try

Array Hurdle Example (●●◌)

`array_multiplied.c`

Write a C program that reads integers into an array from terminal until a number is entered which when multiplied by another number previously entered results in 56. E.g.

```
./array_multiplied
2
3
28
28 * 2 = 56
```

Array Hurdle Example (●●◌)

`array2d_min_row_sum.c`

Write a C program that finds the sum of the minimum numbers in each row in a 2D array.

Array Hurdle Example (●●◌)

`array_odd_even.c`

Write a C program that reads integers from standard input until it reads a negative integer.

It should then print the odd numbers on one line and then print the even numbers on the next line.

You can assume a maximum of 1000 integers are read before a negative integer is read

Array Hurdle Example (●●◌)

```
$ ./array_odd_even
1
2
3
2
-42
Odd numbers were: 1 3
Even numbers were: 2 2
```

This is an exercise for you to try

Array Hurdle Example (●●◌)

array_subarray.c

Write a C program to find the largest sum of contiguous subarray in an array.

New: This is an exercise for you to try

Array Example (●●◌)

array_isogram.c

An isogram is a word, in which no letter of the alphabet occurs more than once. Write a C program that reads in strings until Ctrl+D, and checks whether each string is an isogram.

New: This is an exercise for you to try

Debug Example (●◌◌)

`debug.c`

The following code is meant to join two strings together and form one string. For example:

```
$ ./debug
$ Enter the first string: Pass
$ Enter the second string: Word
PassWord
```

Command Line Arguments (●●●)

`common_argument.c`

Write a program to print out the most common command line argument and print out the number of occurrences e.g

```
./common_argument hooray no hooray hooray potato
Most common argument is hooray and it occurs 3
times
./common_argument
There are no arguments
```

Array Example (●●●)

`array_substring.c`

Write a C program that reads in words from standard input until Ctrl+D, and checks whether a word read in as a command line argument is a substring in the word. If it appears, print it again. For example:

```
$./array_substring courage age bloom encourage blooming
blooming
```

New:This is an exercise for you to try

Q&A with the tutors

Your chance to hear/ask about some or all of the following

- Exam tips
- What to enrol in next term
- Getting internships and jobs
- Being a tutor

The Final Lecture

- The final Kahoot
- More revision
  - Search and delete - debugging
  - Split folders
  - Two dot hurdles
  - Three dot questions
- Q&A with some course tutors

Thanks and Goodluck! ⭐⭐⭐⭐⭐⭐⭐

A big thanks to all the amazing staff in the course…

the Admins, the Lecture Moderators, Assignment writers, Forum staff, Revision Session staff, Help Session staff and Tutors!

And of course a huge congratulations and thank you to all of the amazing students who have worked hard throughout the term and answered and asked so many great questions!!!

We hope you have learnt a lot!

Reach Out

Content Related Questions:
Forum

Admin related Questions email:
cs1511@unsw.edu.au

Don’t forget to attend Help Sessions
if you need one on one help

Struggling with non-course specific issues?

My Feelings and Mental Health Managing Low Mood, Unusual Feelings & Depression	**Mental Health Connect**	student.unsw.edu.au/**counselling** Telehealth	**In Australia Call Afterhours** **UNSW Mental Health Support Line**	1300 787 026 5pm-9am
	Mind HUB	student.unsw.edu.au/**mind-hub** Online Self-Help Resources	**Outside Australia Afterhours** **24-hour Medibank Hotline**	+61 (2) 8905 0307
Uni and Life Pressures Stress, Financial, Visas, Accommodation & More	**Student Support** **Indigenous Student Support**		– student.unsw.edu.au/**advisors** – nura-gili-centre-indigenous-programs	
Reporting Sexual Assault/Harassment	**Equity Diversity and Inclusion (EDI)**		– edi.unsw.edu.au/**sexual-misconduct**	
Educational Adjustments To Manage my Studies and Disability / Health Condition	**Equitable Learning Services (ELS)**		– student.unsw.edu.au/**els**	
Academic and Study Skills	**Academic Skills**		– student.unsw.edu.au/**skills**	
Special Consideration Because Life Impacts our Studies and Exams	**Special Consideration**		– student.unsw.edu.au/**special-consideration**	