Data Structures and Algorithm COMP9024 25T0

T13A & T14A lab

January 16th 2025 Thursday

Content of today:

1. singly linked list & doubly inked list
2. Assessment 1
3. Q & A

Joffrey Ji (z5450981@ad.unsw.edu.au)

Question 2_1_1

(Counting primitive operations)

The following algorithm

- takes a sorted array A[1..*n*] of characters
- and outputs, in reverse order, all 2-letter words vω such that v≤ω.

```
for all i=n down to 1 do
    for all j=n down to i do
        print "A[i]A[j]"
    end for
end for
```

Count the number of primitive operations (evaluating an expression, indexing into an array). What is the time complexity of this algorithm in big-Oh notation?

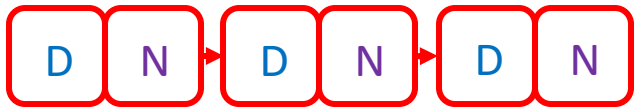
Operation of Singly Linked List

1. Traversing a singly linked list

(head)

2. Append a node to the end of a list

(head)



3. Prepend a node to the start of a list

(head)

4. Inserting a new node to a specific position of a list

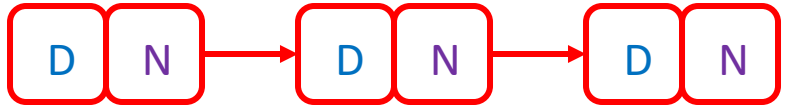
(head)

5. Deleting a node from a list

(head)

6. Updating a node in the list

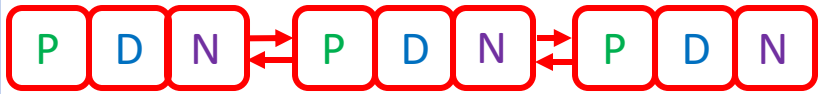
(head)



Operation of Doubly Linked List

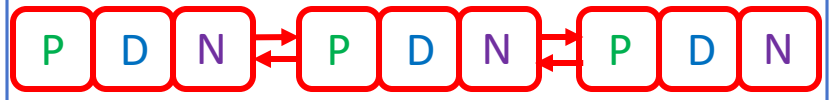
1. Traversing a doubly linked list

(head)



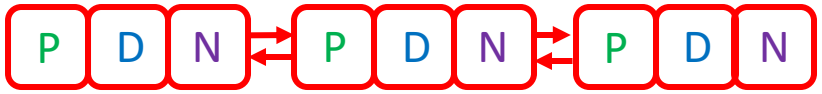
2. Append a node to the end of a list

(head)



3. Prepend a node to the start of a list

(head)

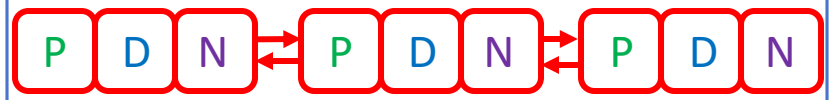


4. Inserting a new node to a specific position of a list

(head)

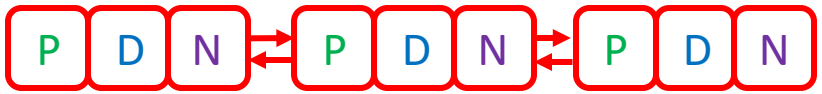
5. Deleting a node from a list

(head)



6. Updating a node in the list

(head)



Question 3

(Algorithms and complexity)

Develop an algorithm to determine if a character array of length contains at least one letter more than once. For example, "repeat" is such a word but "until" is not.

a. Write the algorithm in pseudocode.

b. Analyse the time complexity of your algorithm.

c. Implement your algorithm in C as a function

```
bool hasRepeatedLetter(char A[])
```

that returns `true` if a letter is repeated in string `A`, and `false` otherwise.

Hint: The standard library `<stdbool.h>` defines the basic data type `bool` with the two values `true` (internally encoded as `1`) and `false` (= `0`).

d. Use your solution to Exercise c. to write a program that prompts the user to input a word and checks whether it has a repeated letter. Examples of the program executing are

```
prompt$ ./word
Enter a word: repeat
yes
prompt$ ./word
Enter a string: until
no
```

Hints:

- You may assume that the input consists of lower case letters only.
- You may assume that the input consists of no more than 31 characters (excluding the terminating `'\0'`).
- You can use the standard I/O library function `scanf("%31s",str)` to read a word from the input.
- You may use the standard library function `strlen(char[])`, defined in `<string.h>`, which computes the length of a string (without counting its terminating `'\0'`-character).

We have created a script that can automatically test your program. To run this test you can execute the `dryrun` *program that corresponds to this exercise. It expects to find a program named* `word.c` *in the current directory. You can use dryrun as follows:*

```
prompt$ 9024 dryrun word
```

Data Structures and Algorithm COMP9024 25T0

T13A & T14A lab

January 16th 2025 Thursday

~Questions and Suggestions~

Thanks for listening

Joffrey Ji (z5450981@ad.unsw.edu.au)