Informatics II
Exercise 6

Mar 29, 2021

Goals:

- Practise pointers of primitive types: int, double, char.
- Practise pointers of arrays.
- Study the linked list data structure and implement it in C.
- Practise linked list with a coding task .

Pointers

Task 1. Pointers, values, and addresses.

a) Write a short C program that declares and initializes three variables: double d , int i , and char ch. Next declare and initialize a pointer to each of the three variables: pointer p_d for d, p_i for i, p_ch for ch. Print the following information of d, i, ch, p_d, p_i, and p_ch:

   (a) their values

   (b) their addresses

   (c) memory sizes(in bytes)

   Use the "%p" formatting specifier to print addresses in hexadecimal. You should see addresses that look something like this: "0xbfe55918". The initial characters "0x" tell you that hexadecimal notation is being used; the remainder of the digits give the address itself.

   Use "%f" to print a floating value. Use the *sizeof* operator to determine the memory size allocated for each variable, then use "%lu" to print it .

b) Check the two C functions, *swap_nums* seems to work, but *swap_pointers* does not. Fix it.

```
1  #include <stdio.h>
2
3  void swap_nums(int *x, int *y)
4  {
5    int tmp;
6    tmp = *x;
7    *x = *y;
8    *y = tmp;
9  }
10
11 void swap_pointers(char *x, char *y)
12 {
```

```
13   char *tmp;
14   tmp = x;
15   x = y;
16   y = tmp;
17 }
18
19 int main()
20 {
21   int a,b;
22   char *s1,*s2;
23   a = 3; b=4;
24   swap_nums(&a,&b);
25   printf("a is %d\n", a);
26   printf("b is %d\n", b);
27   s1 = "I should print second";
28   s2 = "I should print first";
29   swap_pointers(s1,s2);
30   printf("s1 is %s\n", s1);
31   printf("s2 is %s\n", s2);
32   return 0;
33 }
```

Task 2. Pointers are widely used to access strings and arrays. In this exercise, we use pointers instead of the `[]` operator to manipulate strings and arrays.

a) Write a program in C to calculate the length of the string **using pointers**.

b) Write a program in C to print a string in reverse **using pointers**.

c) Write a program in C to print the elements of an array in reverse order **using pointers**.

d) Write a C program to multiply two matrix **using pointers**.

Linked Lists

Task 3. We have seen how a linked list works from the lectures. Now let us implement the linked list data structure in C.

a) *void createNodeList(int n)* and *void displayList()* that create and display Singly Linked List (of n nodes).

b) *void insertNode(int num, int pos)* that inserts a new node at the middle of Singly Linked List.

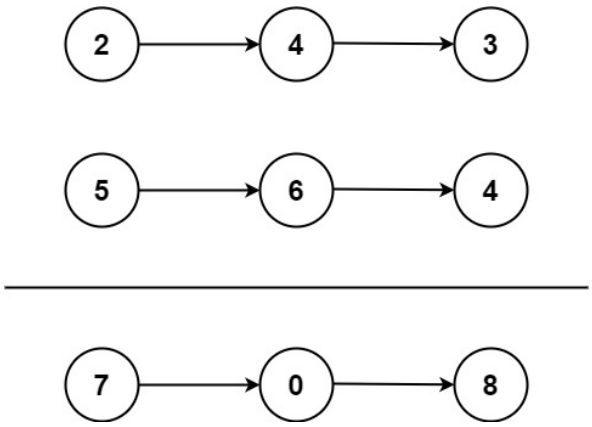
c) *void deleteNode(int pos)* that deletes a node from the middle of Singly Linked List.

d) *int FindElement(int FindElem)* that searches an existing element in a Singly Linked List.

Task 4. You are given two non-empty linked lists representing two non-negative integers. The digits are stored in reverse order, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

Example 1:



Input: $l1 = [2, 4, 3]$, $l2 = [5, 6, 4]$
Output: $[7, 0, 8]$
Explanation: $342 + 465 = 807$.

Example 2:

Input: $l1 = [0], l2 = [0]$
Output: $[0]$

Example 3:

Input: $l1 = [9, 9, 9, 9, 9, 9, 9], l2 = [9, 9, 9, 9]$
Output: $[8, 9, 9, 9, 0, 0, 0, 1]$