ECE 532

TUTORIAL 5 (WEEK 5)

LINKED LIST

Consider the linked list shown in Figure 1. Assume that the nodes are in the usual info-link form. Use this list to answer Exercises 1 through 2. If necessary, declare additional variables. (Assume that list, p, s, A, and B are pointers of type nodeType.)

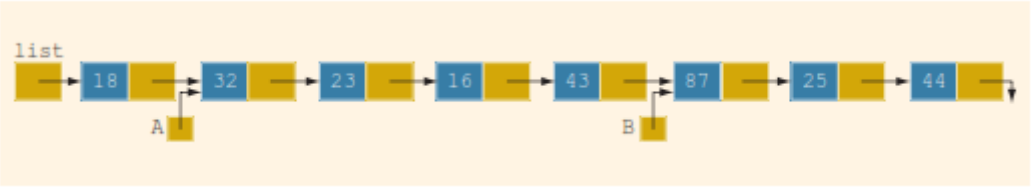


Figure 1

1. If the following C++ code is valid, show the output. If it is invalid, explain why.

```
s = A; p = B;

s->info = B;

 p = p->link;

cout << s->info << " " << p->info << endl;
```

2. If the following C++ code is valid, show the output. If it is invalid, explain why.

```
p = A;

p = p->link;

 s = p;

p->link = NULL;

 s = s->link;

cout << p->info << " " << s->info << endl;
```

3. Show what is produced by the following C++ code. Assume the node is in the usual info-link form with the info of type int. (list and ptr are pointers of type nodeType.)

```
list = new nodeType;

list->info = 10;

ptr = new nodeType;

 ptr->info = 13;
```

```
 ptr->link = NULL;

list->link = ptr;

 ptr = new nodeType;

 ptr->info = 18;

ptr->link = list->link;

list->link = ptr;

cout << list->info << " " << ptr->info << " ";

 ptr = ptr->link;

 cout << ptr->info << endl;
```

4. Show what is produced by the following C++ code. Assume the node is in the usual info-link form with the info of type int. (list and ptr are pointers of type nodeType.)

```
list = new nodeType;

 list->info = 20;

 ptr = new nodeType;

 ptr->info = 28;

 ptr->link = NULL;

list->link = ptr;

ptr = new nodeType;

ptr->info = 30;

 ptr->link = list;

list = ptr;

 ptr = new nodeType;

 ptr->info = 42;

 ptr->link = list->link;

list->link = ptr;

 ptr = list;

while (ptr != NULL)

{
```

```
cout << ptr->info << endl;

ptr = ptr->link;

}
```

5. Assume that the node of a linked list is in the usual info-link form with the info of type int. The following data, as described in parts (a) to (d), is to be inserted into an initially linked list: 72, 43, 8, 12. Suppose that head is a pointer of type nodeType. After the linked list is created, head should point to the first node of the list. Declare additional variables as you need them. Write the C++ code to create the linked list. After the linked list is created, write a code to print the list. What is the output of your code?

a. Insert 72 into an empty linked list.

b. Insert 43 before 72.

c. Insert 8 at the end of the list.

d. Insert 12 after 43.

6. Assume that the node of a linked list is in the usual info-link form with the info of type int. (list and ptr are pointers of type nodeType.) The following code creates a linked list.

```
ptr = new nodeType;

ptr->info = 16;

list = new nodeType;

list->info = 25; list->link = ptr;

ptr = new nodeType;

ptr->info = 12;

ptr->link = NULL;

list->link->link = ptr;
```

Use the linked list created by this code to answer the following questions. (These questions are independent of each other.) Declare additional pointers if you need them.

a. Which pointer points to the first node of the linked list?

b. Determine the order of the nodes of the linked list.

c. Write a C++ code that creates and inserts a node with info 45 after the node with info 16.

d. Write a C++ code that creates and inserts a node with info 58 before the node with info 25. Does this require you to the change the value of the pointer that was pointing to the first node of the linked list?

e. Write a C++ code that deletes the node with info 25. Does this require you to the change the value of the pointer that was pointing to the first node of the linked list?

STACK

7. Suppose the following operations are performed on an empty stack:

push(0);

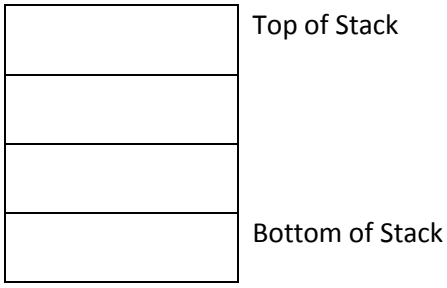
push(9);

push(12);

push(1);

Insert numbers in the following diagram to show what will be stored in the static

stack after the operations above have executed.



8. Suppose the following operations are performed on an empty stack:

push(8);

push(7);

pop();

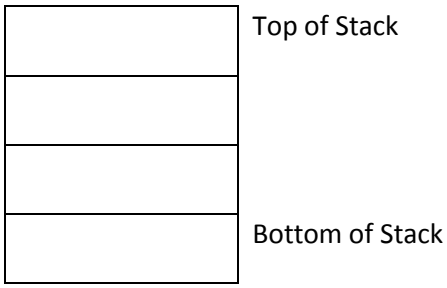
push(19);

push(21);

pop();

Insert numbers in the following diagram to show what will be stored in the static stack

after the operations above have executed.



Stack Applications

9. Evaluate the following postfix expressions using 1) underlining/hand technique and 2)stack algorithm:

a. 8 2 + 3 * 16 4 / - =

b. 12 25 5 1 / / * 8 7 + - =

c. 70 14 4 5 15 3 / * - - / 6 + =

d. 3 5 6 * + 13 - 18 2 / + =

10. Evaluate the following postfix expressions using 1)underlining/hand technique and 2)stack algorithm:

a. -*+ 3 5 6 4
b. + 3 + 4 / 20 4
c. - * - + 3 2 1 / 10 2 - 4 / 6 3