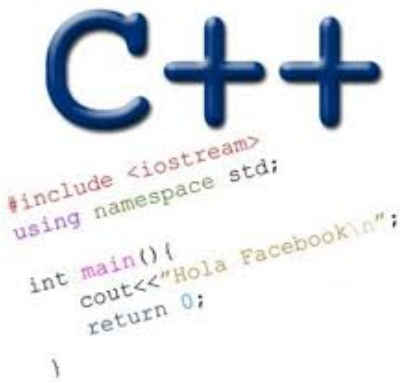
DOUBLE LINKED LISTS

Problem Solving with Computers-I



Dangling pointers and memory leaks

- Dangling pointer: Pointer points to a memory location that no longer exists
- Memory leaks (tardy free):
  - Heap memory not deallocated before the end of program
  - Heap memory that can no longer be accessed

Dynamic memory pitfalls

Memory leaks (tardy free):

Heap memory not deallocated before the end of program

Heap memory that can no longer be accessed

Example

```
void foo(){
     int* p = new int;

}
```

Q: Which of the following functions returns a dangling pointer?

A

```
int* f1(int num){
    int* mem1 =new int[num];
    return(mem1);
}
```

B

```
int* f2(int num){
     int mem2[num];
     return(mem2);
}
```

C Both

D Neither

Pointer pitfalls and memory errors

- **Segmentation faults**: Program crashes because it attempted to access a memory location that either doesn't exist or doesn't have permission to access
- Examples of code that results in undefined behavior and potential segmentation fault

```
int arr[] = {50, 60, 70};

for(int i=0; i<=3; i++){
   cout<<arr[i]<<endl;
}
```

```
int x = 10;
int* p;
cout<<*p<<endl;
```

LinkedList representation in memory

Memory Address	Value
0x8000	0x8008
0x8004	0x8020
0x8008	0x803C
0x800C	0x000A
0x8010	0x8018
0x8014	0x0002
0x8018	0x8030
0x801C	0x0004
0x8020	0x0005
0x8024	0x8014
0x8028	0x0020
0x802C	0x0000
0x8030	0x0003
0x8034	0x8028
0x8038	0x8008
0x803C	0x8000
0x8040	0x8028

Assume that list is a pointer to a LinkedList object (single linked list)

List is stored in the location 0x8008.

Draw the linked-list stored in memory

Double Linked Lists

Array List

Single Linked List

Double Linked List

Implementing a double-linked list

- Define a node in a double linked list
- Write functions to
  - insert a node to the head/tail of the linked list
  - Print all the elements of the list
  - Delete a node with a given value
  - Free the list

Next time

- Recursion