12 Quiz3 /w Ans

T or F: An object should be responsible for changing its own data/

T

The efficiency of the Stack push function is big-O of:

1

What is the "efficiency" of some function? Why the answer is 1? Can someone explain this concept? Thanks!

efficiency here mean, how much of an effort it is for a processor to perform a computing operation. So, pushing onto a stack is a straightforward operation which is accomplished by inserting the data on the top. No traversal or any other manipulation is required to perform an insert (push) onto a stack, just like an array.

So pushing onto stack is simple assignment which is constant and thus O(1) being the complexity (read efficiency).

//Thank you!

The efficiency of the Stack pop function is big-O of:

1

The efficiency of the Stack top function is big-O of:

1

The efficiency of the Linked-List based Stack push function is big-O of:

1

The efficiency of the Linked-List based Stack pop function is big-O of:

1

The efficiency of the Linked-List based Stack top function is big-O of:

1

The efficiency the Linked-List based insert function to insert at the beginning of the list is big-O of:

1

The efficiency of the Linked-List based remove function to remove from the beginning of the list is big-O of:

1

The efficiency of the Linked-List based view function to view at the beginning of the list is big-O of:

1

The efficiency of the Linked-List based view function to view at a selected position in the list is big-O of:

n

The efficiency of the Linked-List based remove function to remove at a selected position in the list is big-O of:

n

The efficiency of the Linked-List based insert function to insert at a selected position in the list is big-O of:

n

The efficiency of the Linked-List based insert function to insert at the end of the list is big-O of:

1 (Is this same for circular and non-circular ?) Yes, since there's always a pointer to the last item inserted it's independent of how many items are in the list

The efficiency of the Linked-List based remove function to remove at the end of the list is big-O of:

1

The efficiency of the Linked-List based view function to view at the end of the list is big-O of:

1

In using a polymorphic generic container, the container changes its behavior based on the

Object it holds

When using polymorphic generic containers, an object _____ when inserted into a container.

Loses its identity

When using polymorphic generic containers, an object _____ when removed from the container.

Regains identity

T or F: When using a polymorphic generic container, the container can manipulate objects only when knowing the types of those objects.

F

T or F: When using a polymorphic generic container, the container can manipulate objects without knowing the types of those objects.

T

T or F: The array based Stack of hw3 is an example of a polymorphic generic container.

F

T or F: The calculator of hw4 is an example of a polymorphic generic container.

F

T or F: The List of hw5 is an example of a polymorphic generic container.

T

T or F: The linked-list based Stack of hw5 is an example of a polymorphic generic container.

T

T or F: The MyRec of hw5 is an example of a polymorphic generic container.

F

T or F: Any object can be inserted in a polymorphic generic container without any restrictions placed on that object.

F

T or F: Any object can be inserted in a polymorphic generic container only when the object satisfies certain constraints.

T

In a linked list, how many Nodes are needed to store an item in the list?

One

T or F: A linked list is a container object made up of 0 or more Nodes.

T

In a linked list, which direction does the pre pointer point?

backwards

In a linked list, which direction does the next pointer point?

forwards

When using a non-circular singly linked list, what is the value of the next pointer in the Node at the end of the list?

null

When using a non-circular doubly linked list, what is the value of the next pointer in the Node at the end of the list?

null

When using a non-circular doubly linked list, what is the value of the pre pointer in the Node at the front of the list?

null

When using a circular doubly linked list, what is the value of the next pointer in the Node at the end of the list?

Address of the node at the front of the list

When using a circular doubly linked list, what is the value of the pre pointer in the Node at the front of the list?

Address of the node at the end of the list

How many NULL pointers are there in a singly linked list with three elements?

One

How many NULL pointers are there in a doubly linked list with three elements?

Two

How many NULL pointers are there in a circular doubly linked list with three elements?

Zero

T or F: When inserting into a doubly linked circular list, the code to insert at the front, end or anywhere in the list is achieved with the same lines of code.

T

T or F: When removing from a doubly linked circular list, the code to remove from the front, end or anywhere in the list is achieved with the same lines of code.

T

T or F: When inserting into a doubly linked circular list at the front or at the end of the list, insertion is done between the same two nodes.

T

T or F: When inserting into a doubly linked circular list, the code to insert at the front, end or anywhere in the list is achieved with three distinct sets of lines of code.

F //is it because sometime you need to change front to point to a new node?

I think it is "you do not use three distinct set of codes" remember the implementation restriction which says we can only call insert_node once when implementing at front/end/sorted, we just update the front pointers differently

T or F: When removing from a doubly linked circular list, the code to remove from the front, end or anywhere in the list is achieved through three distinct sets of lines of code.z

F

T or F: When drawing linked lists, an arrow that points to the bottom of the Node really points to the data field of that Node.

F

What does it mean??? What's bottom?

// i think it is false because a pointer pointing anywhere at the same node is essentially a pointer to the address of the node in memory and not a pointer to the data field of the node

// someone please correct this if i'm wrong

//this question is just really unclear!

// it is. Let's hope someone else can explain this

//meanwhile just try to remember it

// yeah !

T or F: When drawing linked lists, an arrow that points to the side of the Node really points to the middle field of that Node.`

F

T or F: When drawing linked lists, an arrow that points to the top of the Node really points to the upper left corner of that Node.

T (the beginning)

T or F: When drawing linked lists, all arrows that point to the any part of the same Node really point to the upper left corner of that Node.

T

In our linked list of hw5, in what section of memory are all Nodes allocated?

Heap

In our linked list of hw5, in what section of memory are all List objects allocated?

Heap

In our linked list of hw5, in what section of memory are all objects stored in the list allocated?

Heap **Isn't this RTS? Iirc one of the answers for these 3 questions was RTS**

No, I think the objects in the list are nodes, which are stored in HEAP agree with heap.

Professor said heap in discussion

T or F: In our linked list of hw5, the Node objects are all nameless allocated at locations in memory accessible only by starting with a named pointer.

T

In our linked list of hw5, list the two conditions either one of which indicates that the list is empty.

Occupancy == 0; front is null

What is the best one word to describe a list where all insertions and removals occur at the front of the list?

stack

What is the best one word to describe a list where all insertions and removals occur at the end of the list?

Stack // can someone explain how 55 and 56 are the same?? I think this is first in, first out/ last in last out, you can imagine stack 55 is reversed in 56

// yeah but isn't last in last out or fir st in first out, something that a Queue does??yeah,sorry about that, first and last should refer to the position, not the insertsequence.

Front in front out is essentially last in first out, because the front element is the last element you inserted <- thank you to make it clear

What is the best one word to describe a list where all insertions occur at the end of the list and removals occur at the front of the list?

Queue

In our linked list of hw5, the data is stored using _____.

Void pointers//

In our linked list of hw5, the manipulations are achieved through _____.

Function pointers

What are the **4** specific manipulations possible on the items stored in our hw5 list?

copy_func, delete_func, is_greater_than_func, write_func

In inserting 10 items into the list of hw5 through driver1, how many different pointers to MyRec elements are past to List's insert method?

10

In inserting 10 items into the list of hw5 through driver2, how many different pointers to MyRec elements are past to List's insert method?

Zero different, only one pointer

When inserting a MyRec into the list of hw5 through driver1, where is that MyRec object in memory?

Heap

When inserting a MyRec into the list of hw5 through driver2, where is that MyRec object in memory?

RTS is this because in driver2 we place a copy of the object into the list and not the actual object?

Yes, the single copy element is updated each time

＊＊＊Wait in driver2 the copy func uses malloc, so why would it be in rts?＊＊＊

In hw5, which of driver1 or driver2 uses a reusable MyRec object allocated on the Run-Time Stack?

Driver2

The MyRec object structure definition of hw5 was created with what two design goals mentioned in class?

Simple, representative of any future object

What are the four constraint methods of MyRec in hw5?

Copy, delete, is greater than, write

Which of the four constraint methods of MyRec in hw5 **was not found in both** driver1 and driver2?

Copy - absent in driver1 (What the problem is asking for ? "Both"?) <- he's asking for the difference between driver1 and driver2.

Name of method of MyRec in hw5 that was not a constraint method?

new_MyRec //(in driver1

List the 5 layers of driver code for hw5 in order from outermost to innermost.

Driver (user interface) - stack - list - Node - myRec (sample obj)

When the list of hw5 is working correctly, what differences does the user see when running driver1 and driver2?

None

If a user runs driver2, inserts 5 items, and when displayed, the last item inserted is displayed 5 times, what is the likely cause?

The original is stored. Copy is not made. //yes, this is right) what does this mean?

Driver2 copies the element with copy_func and only use the same variable for inserting any element. Thus if you don't make a copy with copy_func everything is pointing to that variable, which would be dthe last eleiment inserted.

"Which is a better test case to test your calc of hw4: A. 1+2 B. 1+1 C. Neither one is better"

A

T or F: One arrow in the drawing of a linked list is achieved through one line of code in your program.

T

When insert_Node's Node pointer parameter is NULL, what does that imply about the item being inserted into the list in hw5?

Inserting into the empty list （I don't know why. When node pointer param is null, why the list is empty?)

Because the node pointer is supposed to be the one before the new item. In a circular linked list only when the list is empty can the item before be null (if there is one node exactly, it will points to itself) I see. Thank you :)

What is the purpose of the MyRec object in hw5?

To test the list

What is the purpose of the MyRec methods in hw5?

To conform with constraint methods so that MyRec can be inserted, to test the list

T or F. The MyRec methods of hw5 have void pointer parameters so these methods can be called passing in objects of different types.

F (then what's the void pointer for???)

(They use void pointers in order to conform to the constraint methods. The constraints use void pointers in order to take in any objects.) //NICE!!

Thank you so much!!

"For the void pointer parameter of MyRec methods of hw5, when will a MyRec pointer be passed to a MyRec method using that parameter:A. Always B. Never C. Sometimes"

A

What is the one method of MyRec from hw5 that is not called polymorphically?

new_MyRec

In what three areas in the code for hw5 are the constraint methods determined?

List struct definition, new list constructor, new stack constructor

Describe the first lines of code in all constraint methods in hw5.

Declaration of a pointer to a true type, and initialize it by casting the parameter to the true type

T or F: When called from the List or Node code in hw5, when a constraint method is called, it is called polymorphically.

T

T or F: When called from the driver1 or driver2 code in hw5, when a constraint method is called, it is called polymorphically.

F is this because driver1 and driver2 is using myRec so it knows the object type? Yes indeed.

T or F: When delete_Node is called from remove_Node, it should delete both the Node and the data it holds.

F

T or F: When delete_Node is called from remove_Node, it should delete the Node but not the data it holds.

T

T or F: When delete_Node is called from remove_Node, it should delete only the data it holds, but not the Node.

F

What are the two ways that delete_Node can be called such that the data it holds won't be deleted?

Either the delete func param is null, or the data inside the Node is null.

T or F: Since the parameter to delete_MyRec is a void pointer, it will only deallocate the data but won't set the originating pointer to NULL.

F

Coding delete_List for hw5, which approach will execute more code, a loop of calls to : A. remove_List B. delete_Node. C. No difference.

A

In a test case of inserting 1 to 5 at the front and 6-10 at the end of the list, what is the best word to describe seeing the list displayed?

Familiar or recognizable

What is the name of the design pattern used in hw5 when the linked-list based Stack was implemented via one line methods calling List methods to perform the underlying implementation?

Facade //(simplified interface to a larger body of code)

What is the best word to describe the algorithm used by each method of the linked-list based Stack of hw5?

delegation

To implement a container using the linked-list of hw5, items are inserted at the _____ of the list and are removed at the _____.

End, front; front, end; one end of the list, the other end

To implement a Stack container using the linked-list of hw5, items are inserted at the _____ of the list and are removed at the _____.

Front, front; end, end; one end of the list, the same end.

Considering a tiered or layered design, List methods called _____ methods, and List methods were called by _____ methods.

Node, Stack

Considering a tiered or layered design, Stack methods called _____ methods, and Stack methods were called by _____ methods.

List, Driver

Considering a tiered or layered design, Node methods called _____ methods, and Node methods were called by _____ methods.

MyRec(func pointer method), List

Not considering validity checks or other error checking, write the one line of linked list based Stack code to implement Push assuming Pop removes from the FRONT of the List.

insert(this_Stack, element, FRONT);

Not considering validity checks or other error checking, write the one line of linked list based Stack code to implement Pop assuming Push inserts at the FRONT of the List.

remove_List(this_Stack, FRONT); (There is no element in pop and top according to stack.c right?) (I think so…) IS THIS RIGHT????

This this like delete the list??

No. remove_List is the method to delete one element from the list. delete_List deletes the whole list.

Not considering validity checks or other error checking, write the one line of linked list based Stack code to implement Top assuming Push inserts at the FRONT of the List.

view(this_Stack, FRONT);

Not considering validity checks or other error checking, write the one line of linked list based Stack code to implement Push assuming Pop removes from the END of the List.

insert(this_Stack, element, END);

Not considering validity checks or other error checking, write the one line of linked list based Stack code to implement Pop assuming Push inserts at the END of the List.

remove_List(this_Stack, END);

Not considering validity checks or other error checking, write the one line of linked list based Stack code to implement Top assuming Push inserts at the END of the List.

view(this_Stack, END);

What is the name of the first parameter to most Stack methods of hw5?

this_Stack

What is the name of the first parameter to most List methods of hw5?

this_List

What is the name of the first parameter to most Node methods of hw5?

this_Node <- agree

Not element???

This is because Gary is trying to port Object oriented programming into C, so compared to this_Node.xxx(...) in java, in C he use xxx(this_Node,...)

T or F: a struct in C defines an object containing only data fields and is similar to a class without methods in Java or C++.

T

"Considering the line of code: this_node->data=(copy_func) ? (*copy_func) (element) : element; What was needed to store the item allocated by main?"

Copy_func needs to be null

"Considering the line of code: this_node->data=(copy_func) ? (*copy_func) (element) : element; What was needed to store an item allocated by the List/Node?"

Copy_func needs to be non-null

T or F: If calling delete_Node from delete_List, the caller wants to delete both the Node and the data.

T

T or F: If calling delete_Node from delete_List, the caller wants to delete the Node but not the data.

F

T or F: If calling delete_Node from delete_List, the caller wants to delete the data but not the Node.

F

"Considering the line of code: if (delete_func && (*npp)->data) (*delete_func) (&((*npp)->data)); List one condition that results in the data not being deleted."

Delete_func is zero; data itself is zero

In the algorithm discussed in class for inserting a Node before the parameter Node in insert_Node of hw5, the first two lines of code attached the Node to the List. What did the next two lines of code do?

Integrate the node to the list

In the algorithm discussed in class for inserting a Node before the parameter Node in insert_Node of hw5, what did the first two lines of code do? The next two lines of code integrated the Node into the List?

Attach the node to the list

When calling free, is the caller giving up access to or authority to access memory or both?

Giving up authority

When assigning a pointer to null, is the caller giving up access to or authority to access memory or both?

Giving up access

In your calculator of hw4, what is the best word to describe the character that decin will "unget"?

Non-digit

In your intopost code of hw4, the values pushed to a stack originate from what two functions?

Setupword, decin

The arrow operator in C and C++ is formed from what two symbols?

->

If you have a pointer called "stup" that points to a Student object, in one line write the code to assign its "number" data field to 123;

stup->number = 123;

"Rewrite the following line of code using preferred syntax:(*pointer).field = 0;"

pointer->field = 0;

"Rewrite the following line of code using preferred syntax:(&object)->field = 0;"

object.field = 0;

Considering a tiered or layered design, each method need to work based on its _____ and _____ values.

Parameters, return

T or F: Considering a tier or layered design, methods of outer tiers are called by methods of inner tiers.

F

T or F: Considering a tier or layered design, methods of inner tiers are called by methods of outer tiers.

T

"T or F: Assuming the below is from correctly compiling code, the following code in Java creates a Student object:void someMethod () { Student stu; }"

F // Can anyone explain this one and the next one? Seems that in the C++ declaration you're creating an object on the stack rather than on the heap; in Java, this isn't allowed so you're only creating a reference and not a heap object. Might be wrong, but that's what I found.

"T or F: Assuming the below is from correctly compiling code, the following code in C creates a Student object: void someMethod () { Student stu; }"

T

In the line, "typedef newtype oldtype;" what is the name of the type being defined?

oldtype

"In the following code, is the ""tag"" mandatory or optional? typedef struct Student { char name[20]; int number; } Student;"

Optional

"In the following code, is the ""tag"" mandatory or optional? typedef struct Student { char name[20]; int number; struct Student * stup;} Student;"

Mandatory. (referring to a student pointer we are defining)

The efficiency of the HashTable to insert in an empty or nearly empty table is big-O of:

1

The efficiency of the HashTable to lookup in an empty or nearly empty table is big-O of:

1

The efficiency of the HashTable to insert in a nearly full table is big-O of:

n

The efficiency of the HashTable to lookup in a nearly full table is big-O of:

n

T or F: The UCSDStudent of hw6 is an example of a polymorphic generic container.

F

T or F: The HashTable of hw6 is an example of a polymorphic generic container.

T

T or F: The Variable of hw6 is an example of a polymorphic generic container.

F

T or F: The SymTab of hw6 is an example of a polymorphic generic container.

T

Considering the main idea of a hash table, where is an item inserted?

Where it belongs

Considering the main idea of a hash table, where do you search for an item?

where we expect to find it

T or F: Hash Tables are often array based.

T

What is the one word to best describe the number that represents the size of a HashTable?

prime

T or F: Hash Tables sizes are often prime so that each space is reachable.

T

What is the word in English to describe the field of an object that is known regardless of whether the operation is lookup or insert?

key

In a typical database, which is the more common operation, insert or lookup?

lookup

In your HashTable of hw6, what is the formula for determining the index of the first location for an item?

ASCII_Sum % table_size

In your HashTable of hw6, what is the formula for determining the increment step for an item?

(ASCII_Sum % (table_size - 1)) + 1

In your HashTable of hw6, what is the formula for determining the next location for an item?

(Current + increment) % table_size

What is the hashing term for the sum of the ASCII codes for the name of an object to insert in the table?

HashCode

What is the hashing term for the list of array indices used to search for a location for an item?

Probe Sequence`

Assume a hash table size of 5, what is the probe sequence for an item with an ASCII sum of 122? (Increment is 3)

2 0 3 1 4

Assume a hash table size of 5, what is the probe sequence for an item with an original location of 3 and an increment of 3?

3 1 4 2 0

For the case of duplicate insertion in a hash table, how should the programmer decide how duplicates should be processed?

Look to the application (Look to the duplicated value already in the table, override its studentnum).

If inserting into a Fairshare hash table, if Eric is in the same location at his first choice that Gary wants as his first choice, who gets the space?

Eric

If inserting into a Fairshare hash table, if Eric is in the same location at his first choice that Gary wants as his second choice, who gets the space?

Gary

If inserting into a Fairshare hash table, if Eric is in the same location at his second choice that Gary wants as his first choice, who gets the space?

Eric

In searching for Gary at his first choice in a Fairshare hash table, Eric is in that location at his second choice. From only that comparison, can you determine that Gary is in the table? A. Gary not there. B. Gary there. C. Can't determine.

C

In searching for Gary at his second choice in a Fairshare hash table, Eric is in that location at his first choice. From only that comparison, can you determine that Gary is in the table? A. Gary not there. B. Gary there. C. Can't determine.

A

I think it is B because Gary is searching for his second choice and he will bump out Eric. The question was asking about lookup.

Can someone explain why the answer is A for question 160? I thought Gary will bump out Eric?

You are correct in that Gary will bump out Eric. However, in the example, Eric stayed in his first choice location which means he was not bumped out which mean Gary does not exist. If Gary did exist, then Eric wouldn't be at his first choice location.
//Thank you~~

In searching for Gary at his first choice in a Fairshare hash table, Eric is in that location at his first choice. From only that comparison, can you determine that Gary is in the table? A. Gary not there. B. Gary there. C. Can't determine.

C

"In your hw6 Locate code in C++, assuming ""element"" is the Base pointer parameter, write the code to determine the hash value: long hashValue = _____;"

(long) *element

"In your hw6 C++ Locate code, assuming ""element"" is the Base pointer parameter, write the code check if table[index] holds a matching item: if (_____)"

***table[index] == *element**

WHY WE NEED * IN FRONT OF TABLE[INDEX]??? THANK YOU!

Because table[index] is dereferenced as table[index] stores the pointer to the object stored at that index in the hashtable

T or F: In your hash table code for hw6, the probe sequence is not stored but is instead the values of "index" as hashing loop executes.

T

T or F: The Fairshare algorithm is an example of Ordered Hashing

T (item has some order of relation)

T or F: The fairshare algorithm is not an example of Ordered Hashing

F

T or F: When using the Fairshare hashing algorithm, an unsuccessful search ends either at an empty space or at an item that has searched less than than the current item.

T

T or F: When using the Fairshare hashing algorithm, an unsuccessful search ends either at an empty space or at an item that has searched longer than the current item.

F

A static method is one called without _____

object

If there are 0 instances of a class, how many instances are there of one of its static data fields?

1

If there are 1000 instances of a class, how many instances are there of one of its static data fields?

1

Is a static data field part of the "sizeof" an object?

No

Is a static data field allocated in adjacent memory to objects of that class?

No

In C, a static method not inside a class is one that can be called by what other methods?

“A `static` function remains visible only in file scope. This is a C feature.” -SO
//so basically static methods can only be called by methods inside the same source file, right?

Any method that exists below the method declaration in the same file

"What is the value of xyz at the end of the fifth execution of the following function?

void func() {static int xyz = 10; xyz++;}

15 // Wouldn’t it be 11 since the value is reset back to 10 every time?

Static variable declarations are executed only once no matter how many times you call the declaration line in one run

" "What is the value of xyz at the end of the fifth execution of the following function?

void func() {int xyz = 10; xyz++;}

11

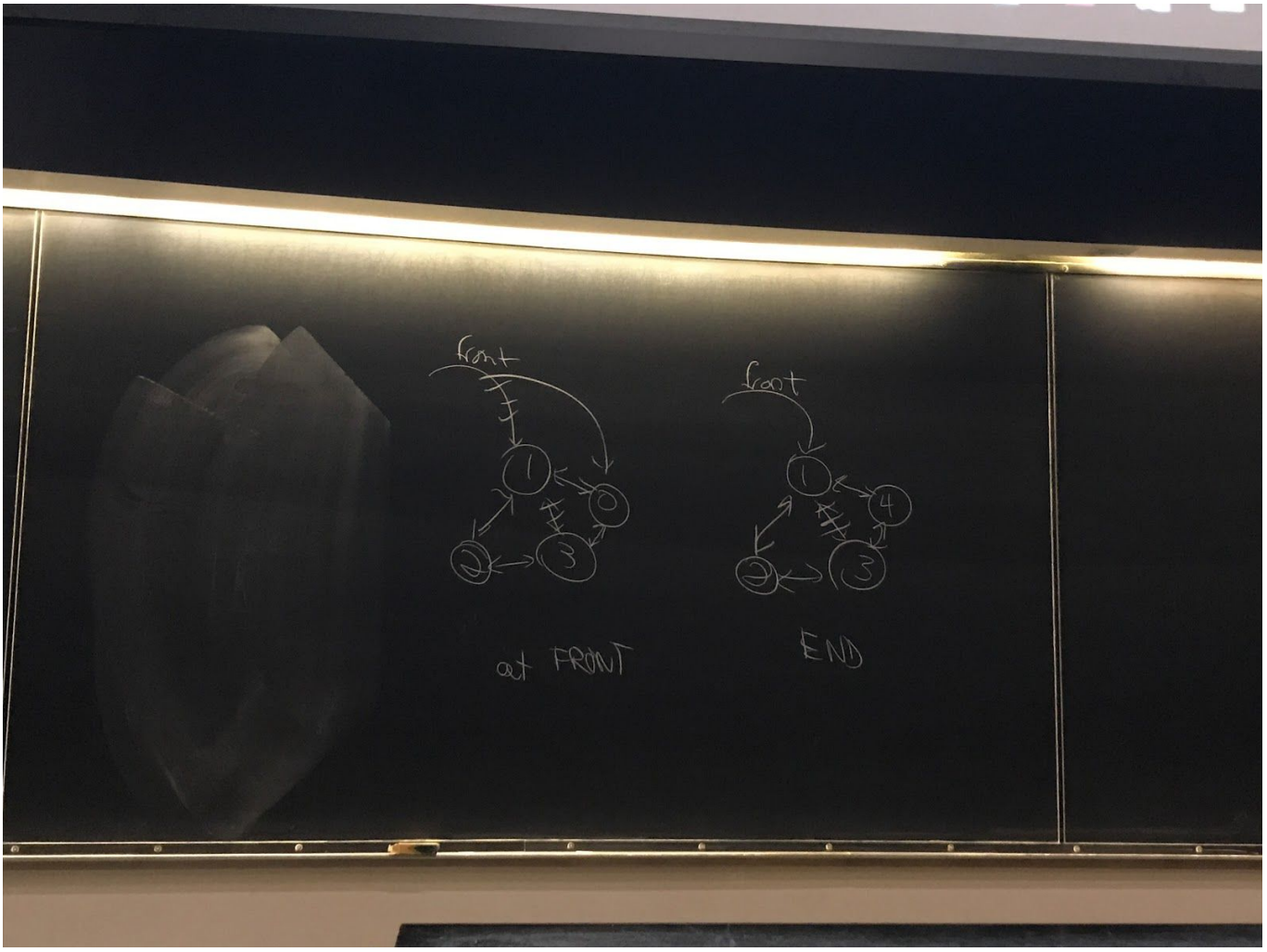
" In C and C++, unless otherwise initialized, what is the initial value of all static variables?

0

Can static methods be accessed by name outside the file in which they are declared?

No

As we did in class, draw two non-detailed drawings of inserting into a circular linked list. One drawing inserts at the front and the other drawing inserts at the end.



Link to an old quizlet set i found:

https://quizlet.com/127249027/learn

For Question 48,

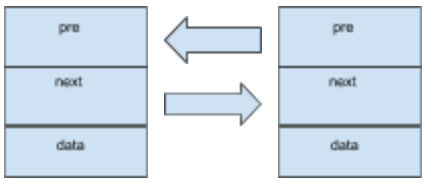
I'm still confused about what it means to have an arrow point to bottom/top/upper left corner.

Can someone help explain this?

-> I think it just means the arrow points to the entire Node.

-> I think the general idea of questions 47-50 refers to the way Gary drew Nodes with a top field of pre, middle field of next, and bottom field of data. Upper left corner refers to node itself.

You can’t point to another node’s fields directly from another node, you can only access its fields



once you are at it. This is why this_list->front->next points to the second node here and not the middle field of the second node. In general, nodes only point to nodes in the circular lists.