Linked List vs Array

Difficulty Level: Rookie

Both Arrays and Linked List can be used to store linear data of similar types, but they both have some advantages and disadvantages over each other.

Following are the points in favour of Linked Lists.

(1) The size of the arrays is fixed: So we must know the upper limit on the number of elements in advance. Also, generally, the allocated memory is equal to the upper limit irrespective of the usage, and in practical uses, upper limit is rarely reached.

(2) Inserting a new element in an array of elements is expensive, because room has to be created for the new elements and to create room existing elements have to shifted.

For example, suppose we maintain a sorted list of IDs in an array id[].

id[] = [1000, 1010, 1050, 2000, 2040, …..].

And if we want to insert a new ID 1005, then to maintain the sorted order, we have to move all the elements after 1000 (excluding 1000).

Deletion is also expensive with arrays until unless some special techniques are used. For example, to delete 1010 in id[], everything after 1010 has to be moved.

So Linked list provides following two advantages over arrays
1) Dynamic size
2) Ease of insertion/deletion

Linked lists have following drawbacks:
1) Random access is not allowed. We have to access elements sequentially starting from the first node. So we cannot do binary search with linked lists.
2) Extra memory space for a pointer is required with each element of the list.
3) Arrays have better cache locality that can make a pretty big difference in performance.

Please also see this thread.

References:

http://cslibrary.stanford.edu/103/LinkedListBasics.pdf

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

56 Comments Category: Arrays Linked Lists

Related Posts:

- Longest Span with same Sum in two Binary arrays
- Count Inversions of size three in a give array
- Find the subarray with least average
- Count triplets with sum smaller than a given value
- Find zeroes to be flipped so that number of consecutive 1's is maximized
- Reorder an array according to given indexes
- Find maximum value of Sum(i*arr[i]) with only rotations on given array allowed
- Find maximum average subarray of k length

Writing code in comment? Please use code.geeksforgeeks.org, generate link and share the link here.

56 Comments GeeksforGeeks 1 Login

Recommend 2 Share Sort by Newest

Join the discussion…

Divya • 2 days ago

Arrays are used for continous storage of data And where the data needs to be accessed randomly. Linked lists are used where insertion or deletion of data is done frequently. In arrays, it is expensive to create room and shift previous ones. In linked lists, pointers of each node occupies space . Taking all these into account, Which has better performance ? Linked lists or arrays?

• Reply • Share ›

Nikita • a month ago

public static void orderZigZag(int[] a) {
for(int i = 0; i<a.length -="" 1;="" i++)="" {="" if(i%2="" !="0)" {="" if(a[i]<a[i+1])="" {="" int="" temp="a[i];" a[i]="a[i+1];" a[i+1]="temp;" }="" }="" else="" {="" if(a[i]="">a[i+1]) {
int temp = a[i];
a[i] = a[i+1];
a[i+1] = temp;
}
}
}
System.out.println(Arrays.toString(a));
}

• Reply • Share ›

Suyash Srivastava • 2 months ago

How to search for by array in a linked list?

• Reply • Share ›

Kishshuuu ➚ Suyash Srivastava • 24 days ago

just check while condition i mean
while(start!=NULL)
// AND now check if condition
if(start->data==key) //key is searching element
element found
//otherwise
start=start->next

not found

• Reply • Share ›

sindhz • 2 months ago

What is true about Linked List?
a) Linked list can't be sorted
b) It must be created using dynamic data allocation
c) No random access is allowed in linked list
d) None of the above

I know that No random access is allowed in linked list and about dynamic data allocation.

Here what comes the correct answer...
b?c?

• Reply • Share ›

Aniket Bhanawase sindhz • a day ago

answer should be option c coz u can create linked list using array data structure also..

• Reply • Share ›

Aman sindhz • 24 days ago

Yes, seems like B and C

• Reply • Share ›

Suyash Srivastava sindhz • 2 months ago

I think random access would be considered as allowed....
If we search for a node,we can access it.

• Reply • Share ›

Kishshuuu Suyash Srivastava • 24 days ago

no
linked list Random access is not allowed.
We have to access elements sequentially starting from the first node. So we cannot do search with linked lists.

• Reply • Share ›

Sindhu Suru • 3 months ago

i mean to ask what is the reason for linked list implementation even though we have an array equation??

• Reply • Share ›

Sindhu Suru • 3 months ago

why mainly linked list is used than array??

• Reply • Share ›

Aman ➚ Sindhu Suru • 24 days ago

Array has fixed size. Linked List can grow / shrink on demand.
Even if we create an array of big enough size once, then inserting & deleting elements in mid of array is very costly as other elements need to be shifted.

• Reply • Share ›

KD Singh • 3 months ago

2) Extra memory space for a pointer is required with each element of the list.
3) Arrays have better cache locality that can make a pretty big difference in performance.

Please also see this thread.

"this" link is broken

• Reply • Share ›

Sexy_coder • 3 months ago

"Arrays have better cache locality that can make them better in terms of performance"---
What does it mean by that ?

• Reply • Share ›

Swaglord ➚ Sexy_coder • 3 months ago

According to the concept of Spatial Locality, if the data in location 'n' is accessed, there's a high chance that the data in location 'n+1' will be accessed soon. Therefore, several contiguous memory blocks are loaded onto the machine's cache to speed up the data access. Since arrays are stored in a contiguous manner, they are best suited for this and thereby improving performance.

2 • Reply • Share ›

Mandla Nkululeko Nkosi • 5 months ago

what are the main functions of linked lists

• Reply • Share ›

Shantanu Madane ➚ Mandla Nkululeko Nkosi • 5 months ago

The main function of linked is its dynamic allocation of elements at runtime

1 • Reply • Share ›

krishna kumar nokha • 6 months ago

can anyone tell me

"Arrays have better cache locality that can make them better in terms of performance " what it's exact meaning

• Reply • Share ›

anna_bond ➚ krishna kumar nokha • 5 months ago

It means a chunk of space as a whole is allocated to an array, all elements will be located in that chunk. Where as in link list memory location of each element is independent they are allocated from the heap, this many hinder the performance

1 • Reply • Share ›

BeautifulCode • 6 months ago

Please also see this thread.

The link does not have anything.Please fix it

2 • Reply • Share ›

rory • 6 months ago

Some have stated that linked lists are much better than arrays. If that is always a valid statement, then why are arrays used at all?

• Reply • Share ›

shruthi reddy rory • 4 months ago

another example can be to maintain values that never change or rarely change like states in india..so it wouldnt be meaningful to use linked list since we know they are fixed..

1 • Reply • Share ›

rahul singh rory • 5 months ago

arrays have limitations on space sometimes..you don't know how long an array you may have to consider for storing the data.but in linked list you can add nodes as much as you need

• Reply • Share ›

KURT WAGNER rory • 6 months ago

thats correct Prince... any element of an array can be accesed at O(1) which is not the case with linked list.

• Reply • Share ›

Prince Bharti rory • 6 months ago

due to access time.

2 • Reply • Share ›

poorva • 9 months ago

what is cache locality??how is it better in array?

1 • Reply • Share ›

Abhinav Rana poorva • 9 months ago

CACHE LOCALITY please read the CO for better knowledge

Basically the memory location close to the one being used by cpu gets frequently used so Array store elements in contiguous manner making it easy and faster to load the elements rather than referring back to Main memory then loading the block of the location of LL's next element and then using the data.This again and again referring to MM and back to even HDD takes a lot of time hence
Arrays have better cache locality that can make a pretty big difference in performance

5 • Reply • Share ›

Guest • a year ago

what can be maximum size of an array ? I mean
int array[size];
is there any limit on size of array?

1 • Reply • Share ›

Kartik Nagpal ➚ Guest • a year ago

int array[size]; // allocate array on stack

Stack is basically a block of memory of fixed size. Trying to allocate a stack variable larger than the size of the stack will cause a stack overflow, which is what the segfault is caused by.

Often the size of the stack is 8MB, so, on a 64-bit machine, assuming int takes 4 bytes int array[2000000] has size 4*2000000 < 8MB (the stack is "safe"), but int max[4000000] has size 4*4000000 > 8MB, so the stack overflows and the program segfaults.

On the other hand, dynamically allocating the array with malloc puts the memory into the heap, which is basically unbounded (4GB on a 32-bit machine, 17,179,869,184GB on a 64-bit machine).

29 • Reply • Share ›

shruthi reddy ➚ Kartik Nagpal • 4 months ago

is that GB in 64-bit or MB?

• Reply • Share ›

Aman Raj ➚ shruthi reddy • 4 months ago

GB

• Reply • Share ›

gansai • a year ago

If there are frequent searches done on data, then it would be preferable to use arrays. If there are lot of insertions/deletions, then linked list. In the production code, has anyone come across a kind of implementation which stores data into array for benefit of search

come across a kind of implementation which stores data into array for benefit of search operations and later when a modification is done, it is done in LL and later conversion from LL to array is done?

5 ▲ ▼ • Reply • Share ›

Kartik Nagpal ➦ gansai • a year ago

In production, it is not efficient to switch between linked list and arrays on same set of data. Consider O(n) extra space and O(n) extra steps while copying.

Instead, when deciding on a data structure for a particular application all operations required to be performed are considered.

In your example above, you need insertions/deletions and search operations to be efficient on the same data structure.

A hash_map can provide:
Insertion: O(1) expected, O(n) worst case
Lookup: O(1) expected, O(n) worst case
Deletion: O(1) expected, O(n) worst case

A balanced binary search tree, such as R-B Tree can provide:
Insertion: O(log n)
Lookup: O(log n)
Deletion: O(log n)

These two data structures are much better choice in production rather than jumping from one data structure to another.

21 ▲ ▼ • Reply • Share ›

LNR ➦ Kartik Nagpal • 7 months ago

Great answer :-)

▲ ▼ • Reply • Share ›

nohappy singh • 2 years ago

Arrays have better cache locality that can make a pretty big difference in performance. .. what does this mean??

▲ ▼ • Reply • Share ›

Tushar Gautam ➦ nohappy singh • 6 months ago

Read this answer http://stackoverflow.com/a/112...
Knowing how branch prediction works and the fact that arrays are stored as contiguous block of elements whereas LL are not, might clear your doubts very well! :)

1 ▲ ▼ • Reply • Share ›

Ankit Singh ➚ Tushar Gautam • 6 months ago

I really got confused with that answer. Can you explain it in a easy way?

1 ^ | v • Reply • Share ›

Tushar Gautam ➚ Ankit Singh • 5 months ago

@Ankit Singh A very simple example. Predict the next number in the series given below:
Series A -> 1,2,3,4,5,6,7, ?
Series B -> 2,10,-3 ,-29,3,?

Which one was more easy to predict? I hope you've understood now! :)

2 ^ | v • Reply • Share ›

Ankit Singh ➚ Tushar Gautam • 5 months ago

Thanks !!!!

^ | v • Reply • Share ›

acodebreaker ➚ nohappy singh • 7 months ago

memory addresses which are near to each other are faster to access

^ | v • Reply • Share ›

johnkru ➚ nohappy singh • 2 years ago

To add to what KX said, any time you load a memory address into CPU you also store this address into cache for a faster access if this specific address will be used again. So here, an array (static) has a better cache Hit if an address is requested again than a linked list that has a dynamic addressing.

^ | v • Reply • Share ›

Vaibhav Bajpai ➚ johnkru • a year ago

Thanks for your answer johnkru. But suppose we have accessed arr[3] just now. Now I want to access arr[4]. How the principle which you said above would help in accessing arr[4]. According to you the cache would have stored the address of arr[3]. But how would it help in accessing arr[4].

Suppose somehow it does. Then what's the difference with Linked list. If we access an element in a LL, then what would be the different thing in accessing the next element as compared with the array.

I just want to know about how this cache locality principle works.

^ | v • Reply • Share ›

mukund pb ➚ Vaibhav Bajpai • a year ago

When any address has to be brought in cache, we also get a series

of other neighboring address in addition to it,
For example is &arr[3] is 0x1000008
we may get address from 0x1000000 to 0x1000010

So, here if somebody wants arr[4] or arr[2] ,he already has that in the cache and memory lookup is prevented.

Same is not true in case of LL, since two consecutive nodes in LL does not have any address correlation and might as well be totally apart, so caching local address does not help in case of LL.

Hope this helps.

7 ▲ ▼ • Reply • Share ›

Rohit ➦ Vaibhav Bajpai • a year ago

reaching to the next element is more quicker (in case of array) than reach any other element in the memory (in case of linked list).

▲ ▼ • Reply • Share ›

kx ➦ nohappy singh • 2 years ago

I think it means that arrays have better performance compared to linked lists, because the elements are co-located in memory. Elements in a linked list might be scattered around, and thus, fetching them won't be as efficient.

10 ▲ ▼ • Reply • Share ›

Deepak Singh • 2 years ago

easy at all... :)

▲ ▼ • Reply • Share ›

Kunal Pandey • 3 years ago

I like

▲ ▼ • Reply • Share ›

Rahul Sundar • 3 years ago

This is the only easy post of all...lol

```
/* Paste your code here (You may delete these lines if not writing code) */
```

4 ▲ ▼ • Reply • Share ›

LoneShadow • 4 years ago

Arrays have better cache locality that can make a pretty big difference in performance.

▲ ▼ • Reply • Share ›

gaurav ➚ LoneShadow • 2 years ago

can you please make this cache locality point clear.

3 ^ | v • Reply • Share ›

Load more comments

Subscribe Add Disqus to your site Add Disqus Add Privacy

 Contact Us! About Us! Advertise with us!