

1027期中review

2025年10月27日 上午 09:08

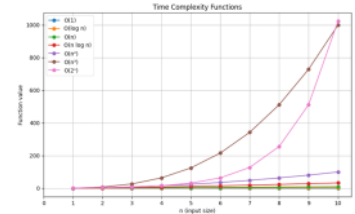
Basic Concept

DS

ADT: specification of operation for a data structure

complexity \leftarrow Time performance evaluation
space

Pointer Manipulation



Data structure

get value by index 0 1 2 3 4 5 memory continuous

- Array

random access

create \leftarrow dynamic malloc realloc ~~temp~~ realloc(array, size)
static \rightarrow int a[5] pointer

insertion $O(1)$

deletion $O(1)$ algorithm \leftarrow selection sort
bubble sort
insertion sort

Search $O(n)$

```

#include <stdio.h>
#include <stdlib.h>

int main() {
    int n;
    int *array;
    int n = 10;

    // Allocate memory for n integers
    array = (int *) malloc(n * sizeof(int));
    if (array == NULL) {
        printf("Memory allocation failed\n");
        return 1;
    }

    // Print the starting memory address
    printf("Initial memory address: %p\n", (void *)array);

    // Insertion elements
    for (int i = 0; i < n; i++) {
        array[i] = i + 1;
    }

    // Print elements
    printf("Initial array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", array[i]);
    }
    printf("\n");

    // Free memory
    free(array);
    return 0;
}

// Double the size
n = n * 2;
array = (int *) realloc(array, n * sizeof(int));
if (array == NULL) {
    printf("Reallocation failed\n");
    return 1;
}

// Print the new memory address
printf("After realloc memory address: %p\n", (void *)array);

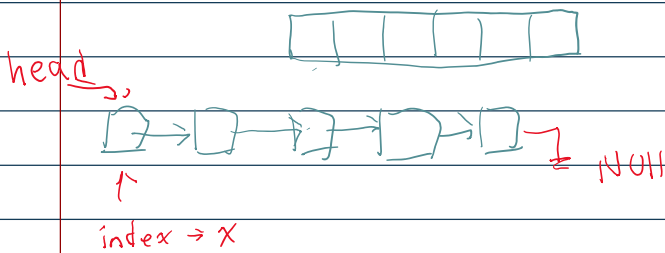
// Insertion new elements
for (int i = n/2; i < n; i++) {
    array[i] = i + 1;
}

// Print all elements
printf("Revised array: ");
for (int i = 0; i < n; i++) {
    printf("%d ", array[i]);
}
printf("\n");

// Free memory
free(array);
return 0;
}

```

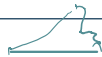
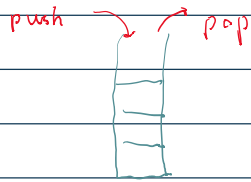
- Linked list



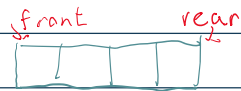
start \rightarrow head \leftarrow data
pointer to next
end \rightarrow NULL

insertion array $O(n) \rightarrow O(1)$ Linked List
deletion $O(n) \rightarrow O(1)$ random access
(due to data shift)

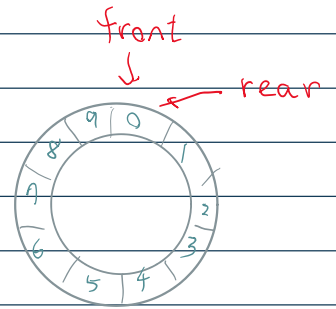
- Stack



- Queue



linear \rightarrow circular



ADT

basic {
 create
 push
 pop

basic {
 create
 enqueue
 dequeue

status {
 isEmpty
 isFull

status {
 isEmpty
 isFull