

# Section A Question1

- (a) Define data structure. Explain its significance in computer science.
- (b) What are the disadvantages of the Linked List over Array?
- (c) When implementing a queue using a singly linked list. Should the front of the queue be placed at tail/head of the list? Explain your reason.
- (d) Define stack data structure. Discuss the role of the stack as recursion call.

- A关键词: organizing data; in memory/computer
- C答案唯一, 从dequeue复杂度解释即可得满分。

# Section B Question 1

- Assuming you have a list contains 300 characters. Analyze the memory needed to save this list into the memory:
- (a) Sequential storage (array)
- (b) Chain storage (single linked without using chunk
- (c) Chain storage (single linked list with chunk of size 30) is use
- (d) Analyze the advantage and the draw back of using array compare to linked list with and without chunk.

- D 从memory角度, deletion和insertion讲即可。

## Question 2

$$A = \begin{Bmatrix} 0 & 1 & 1 & 1 & 1 \\ 4 & 3 & 1 & 1 & 1 \\ 5 & 0 & 1 & 1 & 1 \\ 4 & 1 & 2 & 1 & 1 \\ 7 & 3 & 0 & 4 & 2 \end{Bmatrix}$$

- (a) If the address of the first element  $A[0][0]$  is at the byte 64, show how the matrix will be saved in the memory if using row major order
- (b) Compare the matrix using suitable representation and show the shape after compression.
- (c) Compare the memory needed to save the matrix before and after the compression.

# Section C Question 1

- prefix : \* 9 + 32 - /8 2 3
- 填:

Input	Process	Stack

- Assuming you have an empty stack S and queue Q as shown in Figure 1. Analysis the function in Figure 2 and show (draw) how the queue Q and stack S will look logically after each loop of this code. Show the final output and what is the general purpose of this code.

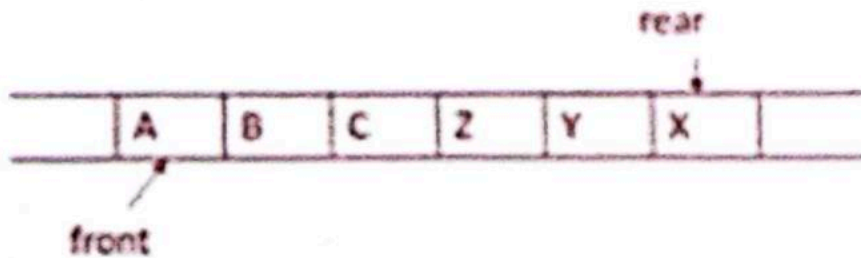


Figure 1: Queue Q

```
void fun(Queue *Q, Stack *S)
{
    while (!IsEmpty(Q))
    {
        push(S, dequeue(Q));
    }

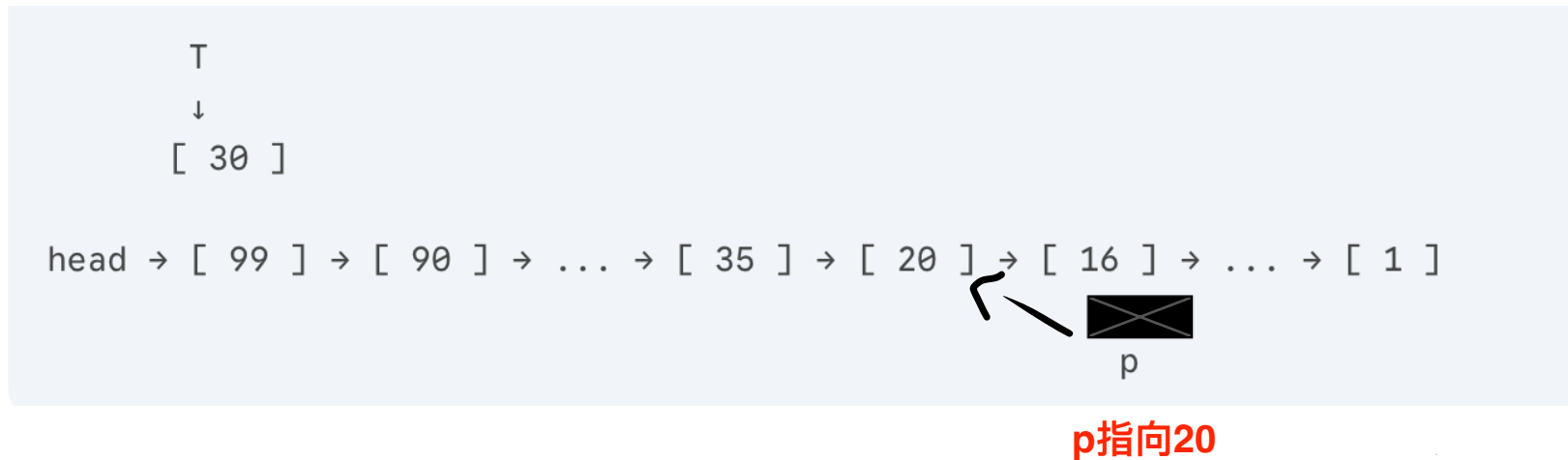
    while (Top(S) != '*')
    {
        enqueue(Q, pop(S));
    }

    while (!IsEmpty(Q))
    {
        push(S, dequeue(Q));
    }

    while (!IsEmpty(S))
    {
        printf(pop(S), " ");
    }
}
```

# Question 3

- Assuming you have a predefined linked list struct contains two fields, **data** (to store integer value) and **next** (pointer to the next node). The head pointer is pointing to the head of the linked list, p is a pointer point to node contains data 20 as in Figure 3 (linked list data are sorted in **descending order**), and there is a pointer T pointing to a new node with data =30.





- a) In the linked list given in Figure 3, complete the function in Figure 4 in your answer sheet to insert node T to the list **before** given node p (you do not have the head of the linked list). [10 marks]

```
Node* insert(Node* T, Node* p) {  
    .....  
    .....  
    .....  
    .....  
    .....  
    return p;  
}
```

(共五行代码要填写)

b) Analyze the functions in Figure 5 line by line and show what is the main purpose of them if these functions are called separately with the head of the linked list in Figure 3. [8 marks]

```
void funA(Node *head)
{
    if (head == NULL) // Line 1
        return;

    funA(head->next); // Line 2
    printf("%d ", head->data); // Line 3
}

int funB(Node *head)
{
    int x = head->data; // Line 1
    head = head->next; // Line 2
    return x; // Line 3
}
```

- (c) What is the big O of funA and funB with assumption that the linked list containing n elements.  
[2 marks]