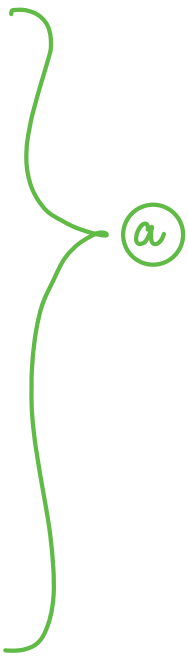
Ques 1 :

```
#include<iostream>

using namespace std ;

class LinkedList
{
	class Node
	{
		public :
		float data ;
		Node *next ;

		Node(float data)
		{
		   this->data = data ;
		   this->next = NULL ;
		}
	} ;

	Node *head = NULL;
	float sum = 0 ;
	int N = 0 ;

	public :

	void Insert(float item)
	{
		// create a new node
		Node *nn = new Node(item) ;

		// linked list is empty || new node to be inserted at beginning
		if(head == NULL || head->data > item)
		{
			nn->next = head ;
			head = nn ;
		}
		 else
		{
			// find the correct position for insertion
			Node *temp = head ;
			while(temp->next != NULL && temp->next->data < item)
			   temp = temp->next ;

			// linking
			nn->next = temp->next ;
			temp->next = nn ;
		}

		 // sum update
```



```
        sum += item ;
        N ++ ;

    }

    void Delete(float item)
    {
        // linked list is empty || new node to be inserted at beginning
        if(head == NULL || head->data == item)
        {
            Node *temp1 = head ; // get the access of the node to be deleted
            head = head->next ; // links set
            delete temp1 ; // free memory
            sum -= item ; // sum update
            N-- ;
            return ;
        }

        Node *temp = head ;
        while(temp->next != NULL)
        {
            if(temp->next->data == item)
            {
               // get the access of the node to be deleted
               Node *temp1 = temp->next ;

               // links set
               temp->next = temp1->next ;

               // free memory
               delete temp1 ;

               // sum update
               sum -= item ;
               N-- ;

               break ;
            }
            temp = temp->next ;
        }
    }

    float getSum()
    {
        return sum ;
    }

    float getAvg()
    {
        return sum / N ;
    }
} ;
```

Ques 2 :

```
#include<iostream>

using namespace std ;

int main()
{
	int arr[] = {2,3,4,5,6,7,8,9} ;
	int N = sizeof(arr) / sizeof(int) ;
	int k = 3 ;

	// One Loop, Time Complexity: O(n), Space Complexity: O(n)
	int na[N] ;
	for(int i = 0 ; i < N ; i++)
	{
		na[i] = arr[(i + N - k) % N] ;
		cout << na[i] << " " ;
	}
	cout << endl ;

	return 0 ;
}
```

Ques 3 :

```
#include<iostream>

using namespace std ;

class Queue
{
	pair<int,bool> *arr ;
	int front ;
	int rear ;
	int size ;
	int cap ;

	public :

	Queue(int capacity)
	{
		cap = capacity ;
		arr = new pair<int,bool>[cap] ;
		front = -1 ;
		rear = -1 ;
		size = 0 ;
	}
```

Logic: Store a boolean variable with every element of queue by utilising 'pair' class.

```
    void push(int item)
    {
        push(item, true) ;
    }

    void push(int item, bool turn)
    {
        if(size == 0)
        {
            front = 0 ;
            rear = 0 ;
        }
        else
            rear = (rear + 1) % cap ;

        arr[rear] = {item,turn} ;
        size ++ ;
    }

    void secondChanceDelete()
    {
        pair<int,bool> temp = arr[front] ;
        front = (front + 1) % cap ;

        if(temp.second == true)
            push(temp.first, false) ;

        size -- ;
    }

    void display()
    {
        for(int i = 0 ; i < size ; i++)
        {
            int idx = (i + front) % cap ;
            cout << arr[idx].first << " " ;
        }
        cout << endl ;
    }
} ;
```