

program 4: linear queues

```

pseudo code: push(): { if (rear = m - 1) { printf("Overflow");
else: if (front == -1) { front = rear = 0
queue[rear] = xc; }

else

```

open [rear] < ;

function "}" of friend

pop(): { if (front == -1)-prints ("underflow"); }
else if (front == rear) { front = rear = -1; }
else front++; } // (exit) slide.

```
display(): { for(int i=front; i=rear; i++)  
    printf("%d", Queue[i]); }
```

code: #include <stdio.h>

#define N 5

```
int queue[10];  
int rear=-1;  
int front=-1;
```

void push(int x){

if (rear == N-1) {

printf ("queue Overflow \n");

return;

۳

else if (front == -1){

~~front = rear = 0;~~

$queue[rear] = x;$

3

} else { queue[+frear] = x; y
} print("pushed".

```

void pop() {
    if (front == -1) {
        printf("queue underflow\n");
        return;
    }
    else if (front == rear) {
        printf("popped element: %d\n", queue[rear]);
        front = rear = -1;
    }
    else {
        printf("popped element: %d\n", queue[rear]);
    }
}

```

```

void display() {
    if (rear == -1) {
        printf("queue is empty\n");
        return;
    }
    printf("queue elements:\n");
    for (int i = front; i <= rear; i++) {
        printf("%d ", queue[i]);
    }
    printf("\n");
}

```

```

int main() {
    int *element;
    printf("\n1: push 2: pop 3: display 4: Exit\n");
    printf("Enter your choice:");
    scanf("%d", &x);
    while (x != 5) {
        switch(x) {
            case 1: {
                printf("Enter an element to push:");
                scanf("%d", &element);
                push(element);
                break;
            }
        }
    }
}

```

case 2: pop();

break;

case 3: "display() function overflows";
break;

case 4: printf("Exiting... \n");

printf("\n 1: push 2: pop 3: display 4: Exit \n");
printf("Enter your choice: ");

Output: 1:Push 2:pop 3:display 4:Exiting
Enter your choice 1

Enter an element to push 1

2:push 2:pop 3:display 4:Exit

Enter your choice 1

Enter an element to push 2

1:Push 2:pop 3:display 4:Exit

Enter your choice 1

Enter an element to push 3

1:Push 2:pop 3:display 4:Exit

Enter your choice 3

1 2 3

Enter an element to push 4

1:Push 2:pop 3:display 4:Exit

Enter your choice 1

Enter an element to push 5

1:push 2:pop 3:display 4:Exit

Enter your choice 2

Enter an element to push 6

an error overflow

1:push 2:pop 3:display 4:Exit

Enter your choice 4

Exiting

program5: circular queue

Date _____	Page _____
------------	------------

pseudo code: push(): { if (front == -1) { front = rear = 0; queue[rear] = x; } else if ((rear + 1) % N == front) { printf("Overflow"); } else { rear = (rear + 1) % N; queue[rear] = x; }

pop(): { if (front == -1) printf("underflow") else if (front == rear) printf("%d", queue[front]); front = rear = -1; } else { printf("%d", queue[front]); front = (front + 1) % N; }

display(): { for (int i = front; i < rear || (i + 1) % N) { printf("%d", queue[i]); } }

```
#include<stdio.h>
```

```
#define N 5
```

```
int queue[N];
```

```
int rear = -1; /* (initially empty) */
```

```
int front = -1; /* (initially empty) */
```

```
void push(int x){ /* (function definition) */
```

```
if ((front + 1) % N == front) { /* (function body) */
```

```
printf("Queue overflow.\n"); /* (function body) */
```

```
return; /* (function body) */
```

```
}
```

```
else if (front == -1) { /* (function body) */
```

```
front = rear = 0; /* (function body) */
```

```
queue[rear] = x; /* (function body) */
```

```
}
```

queue (Implementation)

```
else {  
    rear = (rear + 1) % N;  
    (*queue[rear] = x); // inserting  
} // if (rear <= front) starting ?  
} max(front) = rear // adding  
x = (rear) element
```

void pop() {

```
if (front == -1) // starting (front == front + 1) : error  
printf("queue underflow \n"); //  
return; // front == front
```

```
} // front >= size || back <= front : size
```

```
else if (front == rear) { front =
```

```
printf(" popped element: %d \n", queue[front]);  
(max(i)) // front == rear == i : front + 1 : (size, back  
front = rear = i + 1) + 1 : (size, back)
```

```
else {
```

```
printf(" popped element: %d \n", queue[front]);
```

```
front = (front + 1) % N;
```

```
}
```

void display() {

```
if (rear == -1) {
```

```
printf("queue is empty \n");
```

```
return;
```

```
}
```

```
printf("queue elements: "); // (i + 1) : size, back
```

```
for (int i = front; i != rear; i = (i + 1) % N) {
```

```
printf("%d ", queue[i]); // (i + 1) : size, back
```

```
}
```

```
printf("%d ", queue[i]); // (i + 1) : size, back
```

```
}
```

size = front

size = rear

printf ("%d", queue[rear]);

printf ("\n");

3

int main() {

int x;

int element;

printf ("1:push 2:pop 3:display 4:exit\n");

printf ("Enter your choice: ");

scanf ("%d", &x);

while (x != 4) {

switch (x) {

case 1: {

printf ("Enter an element to push: ");

scanf ("%d", &element);

push(element);

break;

case 2: {

pop();

break;

case 3: {

display();

break;

case 4: {

printf ("Exiting...\n");

return;

default: {

printf ("Wrong choice! Please try again\n");

break;

printf ("Enter your choice: ");

scanf ("%d", &x);

1: push 2: pop 3: display 4: Exit

Enter your choice 1

Enter an element to push 1

Enter your choice 1

Enter an element to push 2

Enter your choice 1

Enter an element to push 3

Enter your choice 1

Enter an element to push 4

Enter your choice 1

Enter an element to push 5

Enter your choice 1

Enter an element to push 6

queue overflow

Enter your choice 2

popped element: 1

Enter your choice 1

Enter an element to push 6

Enter your choice 3

queue element 2 3 4 5 6

Enter your choice 2

popped element: 2

Enter your choice 2

popped element: 3

Enter your choice 2

popped element: 4

Enter your choice 2

popped element: 5

Enter your choice 2

popped element: 6

Enter your choice 2

queue underflow

Enter your choice 4

exiting...