



ใบงานที่ 1

เรื่อง คิววงกลม (Circular Queue)

เสนอ

อาจารย์ ปิยพล ยืนยงสถาวร

จัดทำโดย

นายกฤษฎา วิทยา

65543206041-7

ใบงานนี้เป็นส่วนหนึ่งของรายวิชา โครงสร้างข้อมูลและขั้นตอนวิธี

หลักสูตรวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา

ประจำภาคที่ 1 ปีการศึกษา 2566

คำสั่ง/คำชี้แจง

- แสดงโค้ดโปรแกรมเป็นส่วนๆพร้อมทั้งอธิบาย
- แสดงผลการรันโปรแกรม พร้อมอธิบายการทำงาน
- สรุปผลการทดลอง

ลำดับขั้นตอนการทดลอง

```
12 #include <stdio.h>           //use printf()
13 #include <conio.h>           //use getch()
14 #define N 5                  // Set Max Queue
15 int Q[N];                    // Prepare Queue 0..N-1
16 int x, Qnumber = 0, F = 0, R = 0; // Declare x and initial Qnumber / Front / Rear variable
17 char status = 'N';           // Initial Status = NORMAL
18 char ch;                     // KBD Read variable
```

- **#include <stdio.h>:** เป็นส่วนของโค้ดที่ใช้ระบุว่าโปรแกรมต้องการใช้ฟังก์ชันที่เกี่ยวข้องกับการป้อนออกข้อมูล
- **#include <conio.h>:** เป็นส่วนของโค้ดที่ใช้ระบุว่าโปรแกรมต้องการใช้ฟังก์ชันที่เกี่ยวข้องกับการควบคุมอินพุตทางคีย์บอร์ด getch()
- **#define N 5:** คำสั่งนี้ใช้กำหนดค่าคงที่ (constant) โดยกำหนดค่า N เท่ากับ 5 ซึ่งแทนถึงขนาดสูงสุดของคิว (Queue) ที่สร้างขึ้น
- **int Q[N]:** การประกาศตัวแปร Q เป็นอาร์เรย์ขนาด N ที่ใช้เก็บข้อมูลในคิว โดยใช้ตัวดัชนี (index) เริ่มต้นจาก 0 ถึง N-1
- **int x, Qnumber = 0, F = 0, R = 0:** การประกาศตัวแปร x, Qnumber, F, R ที่ใช้เก็บค่าต่าง ๆ
 - x เป็นตัวแปรที่ใช้เก็บค่าข้อมูลที่จะเพิ่มลงในคิวหรือที่จะถูกลบออกจากคิว
 - Qnumber เป็นตัวแปรที่เก็บค่าหมายเลขคิวของข้อมูลที่ถูกเพิ่มลงในคิว
 - F เป็นตัวแปรที่เก็บตำแหน่งแรกของคิว (Front)
 - R เป็นตัวแปรที่เก็บตำแหน่งสุดท้ายของคิว (Rear)
- **char status = 'N':** ประกาศตัวแปร status เป็นตัวอักษร (char) และกำหนดค่าเริ่มต้นให้เป็น 'N' ซึ่งแทนสถานะของคิวว่าเป็น "NORMAL" (ปกติ)
- **char ch:** ประกาศตัวแปร ch เป็นตัวอักษร (char) ที่ใช้เก็บค่าที่ผู้ใช้ป้อนจากแป้นพิมพ์

```

20 void insertCQ(int y) { // INSERT Function
21     if ((R == F - 1) || (R == N - 1 && F == 1)) { // Check Queue FULL?
22         printf("!!!OVER FLOW!!!...\n");
23         status = 'O'; // set status = OVER FLOW
24     } else {
25         if (R == N - 1) // Loop back of R to 1 if it maximum
26             R = 1;
27         else {
28             R++; // increase R if Normal
29             if (F == 0) // if F is zero set it to 1
30                 F = 1;
31         }
32         Qnumber++; // Increase queue number
33         printf("You are queue number : %d\n", Qnumber);
34         // Display queue number
35         Q[R] = y; // Put data into Queue
36         status = 'N'; // Set status to "NORMAL"
37     }
38 }

```

insertCQ(int y): เป็นฟังก์ชันที่ใช้ในการเพิ่มข้อมูลเข้าสู่ Circular Queue

- ตรวจสอบว่าคิวเต็มหรือไม่ โดยตรวจสอบเงื่อนไข $R == F - 1$ หรือ $(R == N - 1 \ \&\& \ F == 1)$
- ถ้าคิวเต็ม แสดงข้อความ "!!!OVER FLOW!!!..." และตั้งค่า status เป็น 'O' (OVER FLOW)
- ถ้าคิวไม่เต็ม
 - ตรวจสอบว่า R เป็นสุดท้ายของคิวหรือไม่ ถ้าใช่ให้วนกลับมาที่ 1 โดยตั้งค่า $R = 1$
 - ตรวจสอบว่า F เป็น 0 หรือไม่ ถ้าใช่ให้ตั้งค่า $F = 1$
 - เพิ่มค่า Qnumber ที่เก็บหมายเลขคิว
 - แสดงข้อความ "You are queue number : %d" พร้อมแสดงหมายเลขคิว
 - บันทึกข้อมูลลงในคิวที่ตำแหน่ง R
 - ตั้งค่า status เป็น 'N' (NORMAL)

```

40 int deleteCQ() { // DELETE Function
41     int y;
42     if (F == 0) {
43         printf("\n!!!UNDER FLOW!!!...\n");
44         status = 'U'; // set STATUS = "UNDER FLOW"
45     } else {
46         y = Q[F]; // Get data from Queue
47         Q[F] = 0; // Clear
48         if (F == R) { // Set both to 0 if F and R are same value
49             F = 0;
50             R = 0;
51         } else {
52             if (F == N - 1) // Set F to 1 if F is maximum, otherwise increase
53                 F = 1;
54             else
55                 F++;
56         }
57         status = 'N'; // Set status to "NORMAL"
58         return (y); // Return data
59     }
60 }

```

deleteCQ(): เป็นฟังก์ชันที่ใช้ในการลบข้อมูลออกจาก Circular Queue

- ตรวจสอบว่าคิวว่างหรือไม่ โดยตรวจสอบเงื่อนไข $F == 0$
- ถ้าคิวว่าง แสดงข้อความ "!!!UNDER FLOW!!!..." และตั้งค่า status เป็น 'U' (UNDER FLOW)
- ถ้าคิวไม่ว่าง
 - เก็บข้อมูลที่ตำแหน่ง F ไว้ในตัวแปร y
 - ล้างค่าในคิวที่ตำแหน่ง F
 - ตรวจสอบว่า F และ R เท่ากันหรือไม่ ถ้าใช่ให้ตั้งค่าทั้ง F และ R เป็น 0
 - ถ้า F เป็นสุดท้ายของคิว ให้ตั้งค่า $F = 1$ ไม่งั้นเพิ่มค่า F ขึ้นไปหนึ่ง
 - ตั้งค่า status เป็น 'N' (NORMAL)
 - คืนค่า y (ข้อมูลที่ถูกลบออกจากคิว)

```

62 int DataInQueue() { // Calculate Data waiting in queue
63     int y = 0;
64     if (F != 0 && R != 0) // if not equal then can calculate
65     {
66         if (F ≤ R)
67             y = R - F + 1; // Normal F and R
68         else
69             y = (N - 1) - F + 1 + R; // incase loop of R
70     }
71     return (y);
72 }

```

DataInQueue(): เป็นฟังก์ชันที่ใช้ในการคำนวณจำนวนข้อมูลที่กำลังรออยู่ใน Circular Queue

- ตรวจสอบว่า F และ R ไม่เท่ากับ 0 เพื่อให้สามารถคำนวณได้
- ถ้า F น้อยกว่าหรือเท่ากับ R คำนวณค่า $R - F + 1$ (ข้อมูลที่รอในคิวเป็นปกติ)
- ถ้า F มากกว่า R คำนวณค่า $(N - 1) - F + 1 + R$ (ข้อมูลที่รอในคิวเมื่อตำแหน่ง R กลับเข้ามาในตำแหน่ง 1)

```

74 void ShowAllQueue() { // Display Function
75     int i; // Counter variable
76     printf("N : %d\n", N - 1);
77     printf("Status = %c\n", status); // Display STATUS
78     printf("Data waiting in queue = %d\n", DataInQueue()); // Display Data waiting in queue
79     printf(" F = %d / R = %d\n", F, R); // Display F R
80     for (i = 1; i < N; i++)
81     {
82         printf("%d:%d / ", i, Q[i]); // Display all of data in QUEUE
83     }
84     printf("\n-----\n");
85 }
86

```

ShowAllQueue(): เป็นฟังก์ชันที่ใช้ในการแสดงข้อมูลทั้งหมดใน Circular Queue

- แสดงค่าของ N - 1 (ความสามารถในการเก็บข้อมูลของคิว)
- แสดงสถานะ (status) ของคิว
- แสดงจำนวนข้อมูลที่รอในคิวโดยเรียกใช้ฟังก์ชัน DataInQueue()
- แสดงค่าของตัวชี้ F และ R
- วนลูปเพื่อแสดงข้อมูลที่อยู่ในคิวทั้งหมด โดยแสดงเป็น "หมายเลขคิว:ข้อมูล / "

```

87 int main() {
88     printf("CICULAR QUEUE PROGRAM...\n");
89     printf("=====\n");
90     ch = ' ';
91     while (ch != 'E')
92     {
93         printf("\n[1=INSERT : 2=DELETE E:Exit] : "); // Show MENU
94         ch = getch(); // Wait and read KBD with out ENTER Press
95         switch (ch) // Check ch
96         {
97             case '1':
98                 printf("\nInsert Number : ");
99                 scanf("%d", &x); // Read data from KBD
100                insertCQ(x); // Call INSERTNQ Function
101                ShowAllQueue(); // Display all data in Queue
102                break;
103             case '2':
104                 x = deleteCQ(); // Delete data
105                 printf("\nData from Queue = %d\n", x); // Display it
106                 ShowAllQueue(); // Display all data in Queue
107                 break;
108         } // End SWITCH CASE
109     } // End WHILE Loop
110     printf("\n"); // line feed
111     return (0);
112 } // End MAIN Fn.

```

main(): เป็นฟังก์ชันหลักที่เรียกใช้งานฟังก์ชันและโปรแกรมหลัก โดยมีขั้นตอนดังนี้:

- แสดงข้อความเริ่มต้น "CICULAR QUEUE PROGRAM..."
- ทำงานตามเงื่อนไข `ch != 'E'` ซึ่งหมายถึงให้ทำงานในลูปเมื่อผู้ใช้ไม่ได้เลือก 'E' เพื่อออกจากโปรแกรม ในลูปนี้จะมีเมนูที่แสดงให้ผู้ใช้เลือกเพื่อทำงานต่าง ๆ อ่านค่าอินพุตจากผู้ใช้ด้วยฟังก์ชัน `getch()`
- ตรวจสอบค่าอินพุตที่ผู้ใช้ป้อน และดำเนินการตามเงื่อนไข
- ถ้าผู้ใช้เลือก '1' โปรแกรมจะแสดงข้อความ "Insert Number: " เพื่อให้ผู้ใช้ป้อนข้อมูลที่ต้องการเพิ่มลงในคิว จากนั้นโปรแกรมจะเรียกใช้ฟังก์ชัน `insertCQ(x)` เพื่อทำการเพิ่มข้อมูลลงใน Circular Queue ที่ใส่ค่า `x` ที่ผู้ใช้ป้อนเข้าไป จากนั้นโปรแกรมจะเรียกใช้ฟังก์ชัน `ShowAllQueue()` เพื่อแสดงข้อมูลทั้งหมดในคิวหลังจากมีการเพิ่มข้อมูล
- ถ้าผู้ใช้เลือก '2' โปรแกรมจะเรียกใช้ฟังก์ชัน `deleteCQ()` เพื่อลบข้อมูลออกจาก Circular Queue และโปรแกรมจะแสดงข้อมูลที่ถูกลบออกมา จากนั้นโปรแกรมจะเรียกใช้ฟังก์ชัน `ShowAllQueue()` เพื่อแสดงข้อมูลทั้งหมดในคิวหลังจากมีการลบข้อมูล
- เมื่อผู้ใช้ป้อน 'E' (Exit) โปรแกรมจะสิ้นสุดการทำงาน

Run Program

```
PowerShell
Krits D:\WorkAndProject\C\Data_Structures\Week_03 main pwsh
$ .\Circular_Queue.exe
CICULAR QUEUE PROGRAM...
=====

[1=INSERT : 2=DELETE E:Exit] : _
```

เริ่มโปรแกรมจะมีแสดงข้อความและจะมีการให้ Input ค่า Menu ลงไป โดย 1 คือเพิ่มข้อมูล 2 คือลบข้อมูล E คือออกจากโปรแกรม

```
PowerShell
Krits D:\WorkAndProject\C\Data_Structures\Week_03 main pwsh
$ .\Circular_Queue.exe
CICULAR QUEUE PROGRAM...
=====

[1=INSERT : 2=DELETE E:Exit] :
Insert Number : 10
Youe are queue number : 1
N : 4
Status = N
Data waiting in queue = 1
F = 1 / R = 1
1:10 / 2:0 / 3:0 / 4:0 /
-----

[1=INSERT : 2=DELETE E:Exit] :
Insert Number : 20
Youe are queue number : 2
N : 4
Status = N
Data waiting in queue = 2
F = 1 / R = 2
1:10 / 2:20 / 3:0 / 4:0 /
-----

[1=INSERT : 2=DELETE E:Exit] :
Insert Number : 30
Youe are queue number : 3
N : 4
Status = N
Data waiting in queue = 3
F = 1 / R = 3
1:10 / 2:20 / 3:30 / 4:0 /
-----

[1=INSERT : 2=DELETE E:Exit] :
Insert Number : 40
Youe are queue number : 4
N : 4
Status = N
Data waiting in queue = 4
F = 1 / R = 4
1:10 / 2:20 / 3:30 / 4:40 /
-----

[1=INSERT : 2=DELETE E:Exit] :
Insert Number : 50
!!!OVER FLOW!!!...
N : 4
Status = 0
Data waiting in queue = 4
F = 1 / R = 4
1:10 / 2:20 / 3:30 / 4:40 /
-----
```

INSERT เพิ่มข้อมูล จะเพิ่มข้อมูลเก็บไว้ ถ้าข้อมูลเต็มแล้วจะแสดง !!!OVER FLOW!!!... และ Status เป็น 0

```
PowerShell X + -
Krits D:\WorkAndProject\c\Data_Structures\Week_03 main pwsh
$ .\Circular_Queue.exe
CIRCULAR QUEUE PROGRAM...
=====

[1=INSERT : 2=DELETE E:Exit] :
Insert Number : 10
Youe are queue number : 1
N : 4
Status = N
Data waiting in queue = 1
F = 1 / R = 1
1:10 / 2:0 / 3:0 / 4:0 /
-----

[1=INSERT : 2=DELETE E:Exit] :
Insert Number : 20
Youe are queue number : 2
N : 4
Status = N
Data waiting in queue = 2
F = 1 / R = 2
1:10 / 2:20 / 3:0 / 4:0 /
-----

[1=INSERT : 2=DELETE E:Exit] :
Data from Queue = 10
N : 4
Status = N
Data waiting in queue = 1
F = 2 / R = 2
1:0 / 2:20 / 3:0 / 4:0 /
-----

[1=INSERT : 2=DELETE E:Exit] :
Data from Queue = 20
N : 4
Status = N
Data waiting in queue = 0
F = 0 / R = 0
1:0 / 2:0 / 3:0 / 4:0 /
-----

[1=INSERT : 2=DELETE E:Exit] :
!!!UNDER FLOW!!!...

Data from Queue = 21
N : 4
Status = U
Data waiting in queue = 0
F = 0 / R = 0
1:0 / 2:0 / 3:0 / 4:0 /
-----
```

DELETE ลบข้อมูล จะลบข้อมูลที่รับมาตัวแรกก่อน ถ้าข้อมูลที่ลบไม่มีแล้วจะแสดง !!!UNDER FLOW!!!... และ Status เป็น U

สรุปผลการทดลอง

โปรแกรม Circular Queue นี้เป็นโปรแกรมที่ใช้ในการจัดการข้อมูลในคิว โดยสามารถเพิ่มข้อมูลลงในคิวและลบข้อมูลออกจากคิวได้ โดยใช้เมนูในการควบคุมการทำงานของโปรแกรม เมื่อผู้ใช้เลือกเพิ่มข้อมูล (เลือก '1') โปรแกรมจะรับข้อมูลจากผู้ใช้และเพิ่มข้อมูลลงในคิว และเมื่อผู้ใช้เลือกลบข้อมูล (เลือก '2') โปรแกรมจะลบข้อมูลออกจากคิว โปรแกรมจะทำงานในลูปเมนูจนกว่าผู้ใช้จะเลือกออกจากโปรแกรม (เลือก 'E')