ใบงานที่ 1

เรื่อง คิววงกลม (Circular Queue)

เสนอ

อาจารย์ ปิยพล ยืนยงสถาวร

จัดทำโดย

นายกฤษฎา วิริยา

65543206041-7

ใบงานนี้เป็นส่วนหนึ่งของรายวิชา โครงสร้างข้อมูลและขั้นตอนวิธี

หลักสูตรวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา

ประจำภาคที่ 1 ปีการศึกษา 2566

คำสั่ง/คำชี้แจง

- แสดงโค้ดปรแกรมเป็นส่วนๆพร้อมทั้งอธิบาย
- แสดงผลการรันโปรแกรม พร้อมอธิบายการทำงาน
- สรุปผลการทดลอง

ลำดับดับขั้นการทดลอง

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

```c
void insertCQ(int y) { // INSERT Function
  if ((R == F - 1) || (R == N - 1 && F == 1)) { // Check Queue FULL?
    printf("!!!OVER FLOW!!!...\n");
    status = 'O'; // set status = OVER FLOW
  } else {
    if (R == N - 1) // Loop back of R to 1 if it maximum
      R = 1;
    else {
      R++;          // increase R if Normal
      if (F == 0) // if F is zero set it to 1
        F = 1;
    }
    Qnumber++; // Increase queue number
    printf("Youe are queue number : %d\n", Qnumber);
    // Display queue number
    Q[R] = y;      // Put data into Queue
    status = 'N'; // Set status to "NORMAL"
  }
}
```

insertCQ(int y): เป็นฟังก์ชั่นที่ใช้ในการเพิ่มข้อมูลเข้าสู่ 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)

```c
int deleteCQ() { // DELETE Function
  int y;
  if (F == 0) {
    printf("\n!!!UNDER FLOW!!!...\n");
    status = 'U'; // set STATUS = "UNDER FLOW"
  } else {
    y = Q[F];   // Get data from Queue
    Q[F] = 0;   // Clear
    if (F == R) { // Set both to 0 if F and R are same value
      F = 0;
      R = 0;
    } else {
      if (F == N - 1) // Set F to 1 if F is maximum, otherwise increase
        F = 1;
      else
        F++;
    }
    status = 'N'; // Set status to "NORMAL"
    return (y);   // Return data
  }
}
```

deleteCQ(): เป็นฟังก์ชั่นที่ใช้ในการลบข้อมูลออกจาก 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)

```c
int DataInQueue() { // Calculate Data waiting in queue
  int y = 0;
  if (F != 0 && R != 0) // if not equal then can calculate
  {
    if (F <= R)
      y = R - F + 1; // Normal F and R
    else
      y = (N - 1) - F + 1 + R; // incase loop of R
  }
  return (y);
}
```

DataInQueue(): เป็นฟังก์ชั่นที่ใช้ในการคำนวณจำนวนข้อมูลที่กำลังรออยู่ใน Circular Queue

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

```c
void ShowAllQueue() { // Display Function
  int i; // Counter variable
  printf("N : %d\n", N - 1);
  printf("Status = %c \n", status); // Display STATUS
  printf("Data waiting in queue = %d\n", DataInQueue()); // Display Data waitting in queue
  printf(" F = %d / R = %d\n", F, R); // Display F R
  for (i = 1; i < N; i++)
  {
    printf("%d:%d / ", i, Q[i]); // Display all of data in QUEUE
  }
  printf("\n----------------------------------------------------------\n");
}
```

ShowAllQueue(): เป็นฟังก์ชั่นที่ใช้ในการแสดงข้อมูลทั้งหมดใน Circular Queue

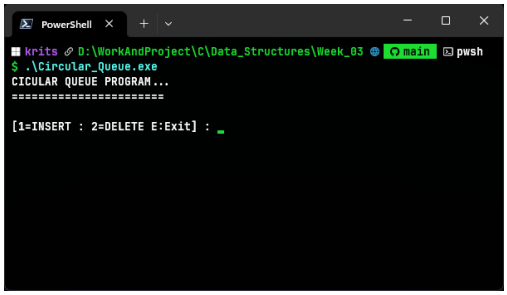
- แสดงค่าของ N - 1 (ความสามารถในการเก็บข้อมูลของคิว)
- แสดงสถานะ (status) ของคิว
- แสดงจำนวนข้อมูลที่รอในคิวโดยเรียกใช้ฟังก์ชั่น DataInQueue()
- แสดงค่าของตัวชี้ F และ R
- วนลูปเพื่อแสดงข้อมูลที่อยู่ในคิวทั้งหมด โดยแสดงเป็น "หมายเลขคิว:ข้อมูล / "

```c
int main() {
  printf("CICULAR QUEUE PROGRAM...\n");
  printf("=======================\n");
  ch = ' ';
  while (ch != 'E')
  {
    printf("\n[1=INSERT : 2=DELETE E:Exit] : "); // Show MENU
    ch = getch();                               // Wait and read KBD with out ENTER Press
    switch (ch)                                 // Check ch
    {
    case '1':
      printf("\nInsert Number : ");
      scanf("%d", &x); // Read data from KBD
      insertCQ(x);     // Call INSERTNQ Function
      ShowAllQueue();  // Display all data in Queue
      break;
    case '2':
      x = deleteCQ();                           // Delete data
      printf("\nData from Queue = %d\n", x); // Display it
      ShowAllQueue();                           // Display all data in Queue
      break;
    }           // End SWITCH CASE
  }             // End WHILE Loop
  printf("\n"); // line feed
  return (0);
} // End MAIN Fn.
```

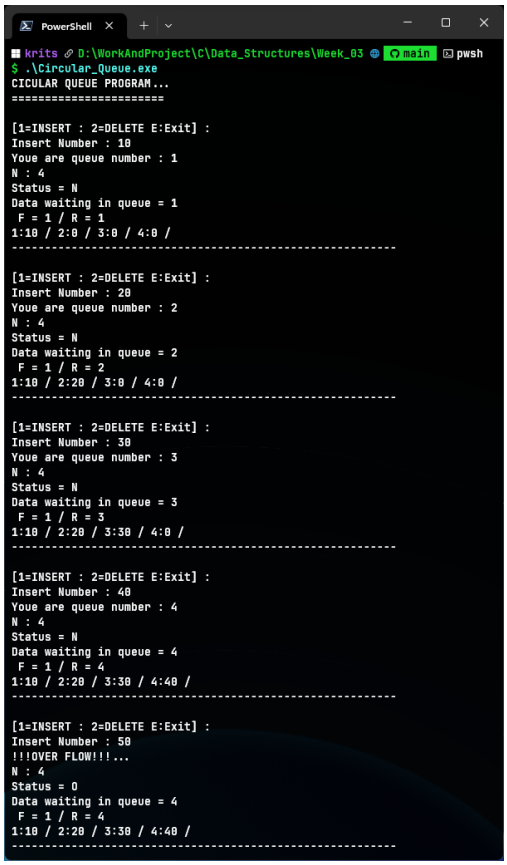
main(): เป็นฟังก์ชันหลักที่เรียกใช้งานฟังก์ชันและโปรแกรมหลัก โดยมีขั้นตอนดังนี้:

- แสดงข้อความเริ่มต้น "CICULAR QUEUE PROGRAM..."
- ทำงานตามเงื่อนไข ch != 'E' ซึ่งหมายถึงให้ทำงานในลูปเมื่อผู้ใช้ไม่ได้เลือก 'E' เพื่อออกจากโปรแกรม ในลูปนี้จะมีเมนูที่แสดงให้ผู้ใช้เลือกเพื่อทำงานต่าง ๆอ่านค่าอินพุตจากผู้ใช้ด้วยฟังก์ชัน 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

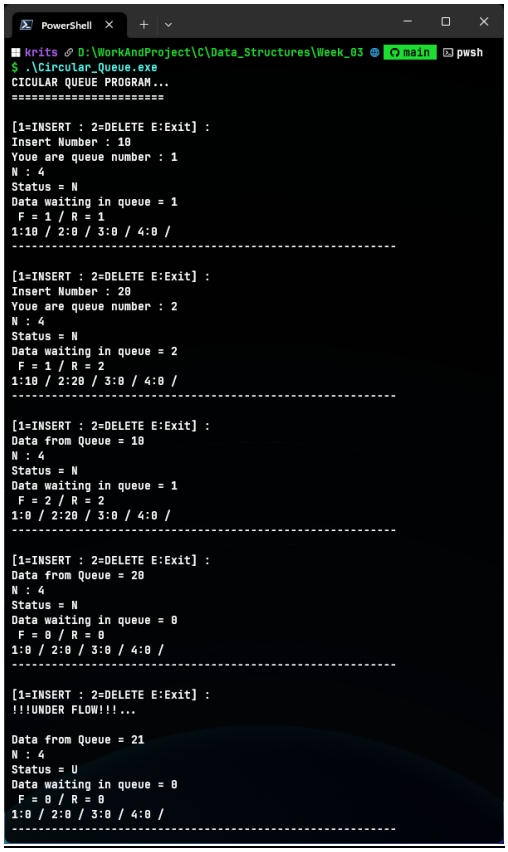
Run Program



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



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



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')