



รายงาน

เรื่อง โปรแกรม QUICK SORT

จัดทำโดย

นายพงษ์พันธุ์ เลาวพงศ์

รหัสนักศึกษา 66543206019-2

เสนอ

อาจารย์ปิยพล ยืนยงสถาวร

รายงานนี้เป็นส่วนหนึ่งของวิชา

ENGCE124

โครงสร้างข้อมูลและขั้นตอนวิธี

(Data Structures and Algorithms)

หลักสูตร วศ.บ.วิศวกรรมคอมพิวเตอร์

สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา ภาคพายัพ เชียงใหม่

ภาคเรียนที่ 1 ปีการศึกษา 2567

รายงาน

เรื่อง โปรแกรม QUICK SORT

จัดทำโดย

นายพงษ์พันธุ์ เลาวพงศ์

รหัสนักศึกษา 66543206019-2

เสนอ

อาจารย์ปิยพล ยืนยงสถาวร

รายงานนี้เป็นส่วนหนึ่งของวิชา

ENGCE124

โครงสร้างข้อมูลและขั้นตอนวิธี

(Data Structures and Algorithms)

หลักสูตร วศ.บ.วิศวกรรมคอมพิวเตอร์

สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา ภาคพายัพ เชียงใหม่

ภาคเรียนที่ 1 ปีการศึกษา 2567

คำนำ

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา ENGCE124 โครงสร้างข้อมูลและขั้นตอนวิธี (Data Structures and Algorithms) หลักสูตร วศ.บ.วิศวกรรมคอมพิวเตอร์ สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา ภาคพายัพ เชียงใหม่ ในระดับปริญญาตรีปีที่ 2 โดยมีจุดประสงค์ในการอธิบายโค้ดของโปรแกรม QUICK SORT รวมถึงอธิบายหลักการทำงานของโปรแกรม QUICK SORT และอธิบายผลลัพธ์การใช้งานโปรแกรม QUICK SORT

ผู้จัดทำรายงานหวังว่า รายงานฉบับนี้จะเป็นประโยชน์กับผู้สนใจ หรือนักศึกษาทุกท่านที่กำลังหาศึกษาในหัวข้อของโปรแกรม QUICK SORT หากมีข้อเสนอแนะหรือข้อผิดพลาดประการใด ผู้จัดทำขอน้อมรับไว้ และขออภัยมา ณ ที่นี้

ผู้จัดทำ

นายพงษ์พันธุ์ เลาหวงศ์

วันที่ 22/09/2567

สารบัญ

	หน้า
คำนำ	ก
สารบัญ	๗
โค้ดของโปรแกรม QUICK SORT พร้อมคำอธิบาย	1
หลักการทำงานของโปรแกรม QUICK SORT	5
ผลลัพธ์การใช้งานโปรแกรม QUICK SORT	12
บรรณานุกรม	15

โค้ดของโปรแกรม QUICK SORT พร้อมคำอธิบาย

```
#include <stdio.h> //ใช้ printf
#include <conio.h> //ใช้ getch
#include <stdlib.h> //ใช้ random
#include <time.h> //ใช้ time

#define MaxData 100 //กำหนดจำนวนข้อมูลสูงสุด

int Data[MaxData];

int i, N;

void PrepareRawData(int N)
{
    int i;

    srand(time(NULL)); //สำหรับการสุ่มหมายเลขที่แตกต่างใน rand()

    for (i=1;i<=N;i++)

        Data[i] = 1 + rand() % 99; //สุ่มหมายเลขที่แตกต่างกันในช่วง 1..99
}

void DispData(int N)
{
    int i;

    for(i=1;i<=N;i++)

        printf(" %2d ",Data[i]);

    printf("\n");
}
```

โค้ดของโปรแกรม QUICK SORT พร้อมคำอธิบาย (ต่อ)

```
void swap(int a,int b)

{

    int temp;

    temp = Data[a];

    Data[a] = Data[b];

    Data[b] = temp;

}

void QuickSort(int f, int r) //ฟังก์ชันแบบเรียกซ้ำ

{

    int f1, r1;

    bool direction;

    f1 = f; r1 = r; //เก็บค่าของ Front และ Rear เดิม

    direction = true;

    while(f != r)

    {

        if(Data[f] > Data[r]) //กรณีการเรียงลำดับจากน้อยไปมาก

        {

            printf("%2d %2d : ", f, r);

            DispData(N);

            swap(f, r);

            printf("%2d %2d : ", f, r);
```

โค้ดของโปรแกรม QUICK SORT พร้อมคำอธิบาย (ต่อ)

```

    DispData(N);

    direction = !direction; //เปลี่ยนทิศทางการเลื่อน pointer

}

if (direction) //เลื่อน r ไปทางซ้ายถ้าเป็นจริง

    r--;

else

    f++; //เลื่อน f ไปทางขวาถ้าเป็นเท็จ

}

printf("k1=[%2d]-----\n", Data[f]);

//กระบวนการทางด้านซ้าย

if((f > f1) && (f - 1 != f1))

    QuickSort(f1, f - 1); //เรียกซ้ำตำแหน่งใหม่ของ F&R

//กระบวนการทางด้านขวา

if((r < r1) && (r + 1 != r1))

    QuickSort(r + 1, r1); //เรียกซ้ำเพื่อกำหนดตำแหน่งใหม่ของ F&R

}

int main()

{

    printf("ASCENDING QUICK SORT\n");

    printf("=====\n");

    N = 12;

```

โค้ดของโปรแกรม QUICK SORT พร้อมคำอธิบาย (ต่อ)

```
PrepareRawData(N);

printf("Raw Data : ");

DispData(N);

printf("Processing Data...\n");

printf(" F R ");

for(i=1;i<=N;i++)

    printf(" (%2d)", i);

printf("\n");

QuickSort(1, N);

printf("-----\n");

printf("Sorted Data : ");

DispData(N); //แสดงข้อมูลที่จัดเรียงแล้ว

getch();

return(0);

} //จบ Main
```


หลักการทำงานของโปรแกรม QUICK SORT

โปรแกรมนี้ใช้เทคนิคการเรียงลำดับแบบ Quick Sort โดยสุมข้อมูลเข้าไปในอาร์เรย์ และจัดเรียงข้อมูลจากน้อยไปมากผ่านการแบ่งอาร์เรย์ย่อยและสลับค่าตำแหน่งต่าง ๆ ผลลัพธ์ที่ได้จะเป็นข้อมูลที่เรียงลำดับจากน้อยไปมาก โดยการทำงานแต่ละฟังก์ชันมีรายละเอียด ดังต่อไปนี้

1. การนำเข้าไลบรารี

```
#include <stdio.h> //ใช้ printf
#include <conio.h> //ใช้ getch
#include <stdlib.h> //ใช้ random
#include <time.h> //ใช้ time
```

ในส่วนของการนำเข้าไลบรารี (#include) จะมีรายละเอียดดังนี้

- <stdio.h> : ไลบรารีนี้ใช้สำหรับฟังก์ชันการรับและแสดงผลข้อมูล เช่น printf() ที่ใช้ในการพิมพ์ข้อความออกทางหน้าจอ และ scanf() ที่ใช้สำหรับการรับข้อมูลจากผู้ใช้
- <conio.h> : ไลบรารีนี้ใช้ในการทำงานกับการอินพุตจากคีย์บอร์ดในรูปแบบที่ง่ายขึ้น เช่น getch() ซึ่งใช้เพื่อรอให้ผู้ใช้กดปุ่มก่อนที่จะดำเนินการต่อ
- <stdlib.h> : ไลบรารีนี้มีฟังก์ชันที่เกี่ยวข้องกับการจัดการหน่วยความจำ การแปลงค่า และการสุ่ม เช่น rand() ที่ใช้สำหรับสร้างค่าตัวเลขสุ่ม
- <time.h> : ไลบรารีนี้มีฟังก์ชันที่เกี่ยวข้องกับเวลาและวันที่ เช่น time() ที่ใช้เพื่อรับค่าชั่วโมง นาที และวินาทีในรูปแบบ timestamp

2. การกำหนดค่าคงที่

```
#define MaxData 100 // กำหนดข้อมูลสูงสุด
```

ในส่วนของการกำหนดค่าคงที่ จะมีรายละเอียดดังนี้

- #define MaxData 100 : การใช้คำสั่ง #define นี้ใช้เพื่อกำหนดค่าคงที่ (constant) ในโปรแกรม โดย MaxData กำหนดค่าที่ 100 ซึ่งเป็นการกำหนดขนาดสูงสุดของอาร์เรย์ Data[] ในโปรแกรม

ค่าคงที่นี้สามารถถูกใช้ในหลายส่วนของโปรแกรม เช่น การวนลูปหรือจัดการข้อมูล เพื่อให้แน่ใจว่าอาร์เรย์ Data[] จะไม่เกินขนาดที่กำหนด (100 ข้อมูล)

3. การประกาศตัวแปร

```
int Data[MaxData];

int i, N;
```

ในส่วนของการประกาศตัวแปร จะมีรายละเอียดดังนี้

- int Data[MaxData] : ตัวแปร Data[] คืออาร์เรย์ชนิดจำนวนเต็ม (int) ที่สามารถเก็บข้อมูลได้มากถึง 100 ตัวเลข (เนื่องจาก MaxData = 100) ซึ่งตัวเลขในอาร์เรย์นี้จะถูกใช้ในกระบวนการสุ่ม, แสดงผล, และเรียงลำดับข้อมูลในโปรแกรม
- int N : ตัวแปร N ถูกใช้เพื่อเก็บจำนวนข้อมูลที่ต้องการให้โปรแกรมจัดการ ตัวอย่างเช่น หากเราต้องการให้โปรแกรมทำงานกับข้อมูล 12 ค่า เราจะกำหนดค่า N=12 เพื่อให้โปรแกรมทราบว่าควรสุ่มและเรียงลำดับข้อมูลกี่ค่า โดยตัวแปร N จึงเป็นตัวแปรสำคัญที่ถูกใช้ในหลายฟังก์ชันเพื่อตัดสินใจจำนวนข้อมูลที่โปรแกรมต้องทำงานด้วย
- int i : ตัวแปร i ถูกใช้ในส่วนของฟังก์ชัน main() เท่านั้น โดยมีหน้าที่ในการวนลูปเพื่อแสดงเลขลำดับของข้อมูลในขั้นตอนการแสดงรายละเอียดการประมวลผลของ Quick Sort

4. ฟังก์ชัน PrepareRawData

```
void PrepareRawData(int N)
{
    int i;

    srand(time(NULL)); //สำหรับการสุ่มหมายเลขที่แตกต่างใน rand()

    for (i=1;i<=N;i++)

        Data[i] = 1 + rand() % 99; //สุ่มหมายเลขที่แตกต่างกันในช่วง 1..99
}
```

4. ฟังก์ชัน PrepareRawData (ต่อ)

ฟังก์ชัน PrepareRawData มีหน้าที่เตรียมข้อมูลดิบโดยการสุ่มตัวเลขเข้าไปในอาร์เรย์ Data[] ขนาด N ซึ่งหลักการทำงานเริ่มต้นจากใช้ฟังก์ชัน srand(time(NULL)); เพื่อสุ่มค่าเริ่มต้น (seed) สำหรับฟังก์ชัน rand() ทำให้ค่าที่สุ่มได้ในแต่ละครั้งมีความแตกต่างกัน จากนั้นจะใช้ for loop ทำการวนซ้ำตั้งแต่ค่า i = 1 จนถึง N เพื่อกำหนดค่าให้กับอาร์เรย์ Data[] แต่ละตำแหน่ง โดยใช้คำสั่ง Data[i] = 1 + rand() % 99; ซึ่งจะสุ่มตัวเลขในช่วง 1 ถึง 99 และนำไปใส่ในอาร์เรย์

5. ฟังก์ชัน DispData

```
void DispData(int N)
{
    int i;

    for(i=1;i<=N;i++)

        printf(" %2d ",Data[i]);

    printf("\n");
}
```

ฟังก์ชัน DispData มีหน้าที่แสดงผลข้อมูลในอาร์เรย์ Data[] ซึ่งใช้ในการแสดงผลข้อมูลทั้งก่อนและหลังการเรียงลำดับ รวมถึงระหว่างขั้นตอนการจัดเรียงด้วย โดยหลักการทำงานเริ่มต้นจาก ใช้ for loop วนซ้ำตั้งแต่ค่า i = 1 จนถึง N เพื่อพิมพ์ค่าของอาร์เรย์ Data[i] ออกมา โดยค่าที่พิมพ์ออกมาจะแสดงในรูปแบบที่เป็นระเบียบ โดยใช้ %2d เพื่อแสดงผลตัวเลขให้มีความกว้างอย่างน้อย 2 ช่อง และมีการเว้นช่องว่างระหว่างแต่ละตัวเลข

6. ฟังก์ชัน swap

```
void swap(int a,int b)
{
    int temp;

    temp = Data[a];

    Data[a] = Data[b];
```

6. ฟังก์ชัน swap (ต่อ)

```
Data[b] = temp;

}
```

ฟังก์ชัน swap ทำหน้าที่สลับค่าของอาร์เรย์ Data[] ที่ตำแหน่ง a และ b กล่าวคือ ฟังก์ชันนี้ถูกใช้ในขั้นตอนของการจัดเรียง เพื่อสลับค่าของตำแหน่งต่าง ๆ ในอาร์เรย์ Data[] เมื่อพบว่ามีค่าไม่ตรงตามลำดับที่ต้องการ (เช่น ถ้าเรียงลำดับจากน้อยไปมาก แต่พบค่าที่มากกว่าทางซ้ายของค่าน้อยกว่า) โดยการทำงานจะเริ่มต้นจาก เก็บค่าของ Data[a] ไว้ในตัวแปรชั่วคราว temp จากนั้นนำค่าของ Data[b] ไปเก็บใน Data[a] สุดท้ายจะนำค่าของ temp (ซึ่งเป็นค่าเดิมของ Data[a]) ไปเก็บใน Data[b]

7. ฟังก์ชัน QuickSort

```
void QuickSort(int f, int r) //ฟังก์ชันแบบเรียกซ้ำ
{
    int f1, r1;

    bool direction;

    f1 = f; r1 = r; //เก็บค่าของ Front และ Rear เดิม

    direction = true;

    while(f != r)
    {
        if(Data[f] > Data[r]) //กรณีการเรียงลำดับจากน้อยไปมาก
        {
            printf("%2d %2d : ", f, r);

            DispData(N);

            swap(f, r);

            printf("%2d %2d : ", f, r);
        }
    }
}
```

7. ฟังก์ชัน QuickSort (ต่อ)

```

        DispData(N);

        direction = !direction; //เปลี่ยนทิศทางการเลื่อน pointer
    }

    if (direction) //เลื่อน r ไปทางซ้ายถ้าเป็นจริง

        r--;

    else

        f++; //เลื่อน f ไปทางขวาถ้าเป็นเท็จ

    }

    printf("k1=[%2d]-----\n", Data[f]);

    //กระบวนการทางด้านซ้าย

    if((f > f1) && (f - 1 != f1))

        QuickSort(f1, f - 1); //เรียกซ้ำตำแหน่งใหม่ของ F&R

    //กระบวนการทางด้านขวา

    if((r < r1) && (r + 1 != r1))

        QuickSort(r + 1, r1); //เรียกซ้ำเพื่อกำหนดตำแหน่งใหม่ของ F&R

    }

```

ฟังก์ชัน QuickSort มีหน้าที่จัดเรียงข้อมูลในอาร์เรย์ Data[] ด้วยการเรียงลำดับแบบ Quick Sort ซึ่งเป็นอัลกอริธึมการเรียงลำดับที่ทำงานแบบเรียกซ้ำ (recursive) โดยการทำงานจะเริ่มต้นจาก การตั้งค่าตัวแปรเริ่มต้น โดยกำหนดตัวแปร f1 และ r1 เก็บค่าตำแหน่งเริ่มต้นของ f (front) และ r (rear) เพื่อใช้ในภายหลัง และตัวแปร direction ใช้กำหนดทิศทางการเลื่อนของ pointerว่าจะเลื่อนไปซ้ายหรือขวา เริ่มต้นจาก true (เลื่อนไปซ้าย) ในส่วนของการทำงานจะมีการเปรียบเทียบ โดยใช้ while(f != r) เพื่อทำการวนซ้ำจนกว่าตำแหน่ง f และ r จะตรงกัน ในแต่ละรอบ ถ้าหากค่า Data[f] มากกว่า Data[r] (กรณีที่เราเรียงลำดับจากน้อยไปมาก) จะทำการแสดงข้อมูลโดยใช้ DispData(N) จากนั้นเรียกใช้ฟังก์ชัน swap(f, r) เพื่อสลับค่าระหว่าง

Data[f] และ Data[r] และเปลี่ยนทิศทางของ pointer โดยสลับค่าของตัวแปร direction ในส่วนของ การเลื่อน pointer โดยถ้าตัวแปร direction เป็น true จะเลื่อนตำแหน่ง r ไปทางซ้าย (r--) ถ้า false จะเลื่อน f ไปทางขวา (f++) ส่วนสุดท้ายจะเป็นการแบ่งอาเรย์ย่อย เมื่อจบการวนลูป ฟังก์ชันจะทำการแบ่งอาเรย์ออกเป็น ส่วนซ้ายและขวา ในส่วนแรกจะเรียกใช้ QuickSort(f1, f-1) สำหรับการจัดเรียงอาเรย์ทางซ้ายมือ และในส่วนที่สองจะเรียกใช้ QuickSort(r+1, r1) สำหรับการจัดเรียงอาเรย์ทางขวามือ

8. ฟังก์ชัน main

```
int main()
{
    printf("ASCENDING QUICK SORT\n");

    printf("=====\n");

    N = 12;

    PrepareRawData(N);

    printf("Raw Data : ");

    DispData(N);

    printf("Processing Data...\n");

    printf(" F R ");

    for(i=1;i<=N;i++)

        printf(" (%2d)", i);

    printf("\n");

    QuickSort(1, N);

    printf("-----\n");

    printf("Sorted Data : ");

    DispData(N); //แสดงข้อมูลที่จัดเรียงแล้ว
```

8. ฟังก์ชัน main (ต่อ)

```
    getch();  
  
    return(0);  
  
} //จบ Main
```

ฟังก์ชัน main เป็นฟังก์ชันหลักที่เป็นจุดเริ่มต้นของโปรแกรม โดยจะมีการทำงานตามลำดับ ดังนี้

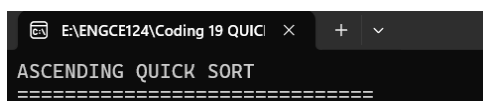
- 8.1 แสดงหัวข้อโปรแกรม "ASCENDING QUICK SORT"
- 8.2 กำหนดค่า N (จำนวนข้อมูล) เป็น 12
- 8.3 เรียกใช้ฟังก์ชัน PrepareRawData(N) เพื่อสุ่มข้อมูลดิบ
- 8.4 แสดงข้อมูลดิบที่สุ่มมาโดยเรียกใช้ฟังก์ชัน DispData(N)
- 8.5 แสดงข้อความ "Processing Data..." และแสดงรายละเอียดการทำงานของ Quick Sort
- 8.6 เรียกใช้ฟังก์ชัน QuickSort(1, N) เพื่อเริ่มต้นกระบวนการจัดเรียงข้อมูล
- 8.7 เมื่อจัดเรียงเสร็จแล้ว แสดงข้อมูลที่เรียงเสร็จโดยเรียกใช้ฟังก์ชัน DispData(N)
- 8.8 รอให้ผู้ใช้กดปุ่ม (ใช้ getch()) ก่อนจบโปรแกรม

ผลลัพธ์การใช้งานโปรแกรม QUICK SORT

โปรแกรม Quick Sort ถูกออกแบบมาเพื่อสุ่มข้อมูลดิบลงในอาร์เรย์ จากนั้นจึงจัดเรียงข้อมูลจากน้อยไปมาก พร้อมแสดงขั้นตอนการทำงานและผลลัพธ์อย่างละเอียด โดยโปรแกรมนี้มีการแสดงผลในหลายช่วง ได้แก่ การแสดงข้อมูลก่อนการจัดเรียง ขณะทำการจัดเรียง และผลลัพธ์สุดท้ายหลังการจัดเรียง โดยผลลัพธ์ที่ได้จะมีดังนี้

1. การแสดงหัวข้อโปรแกรมและข้อมูลเบื้องต้น

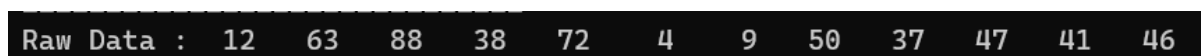
เมื่อเริ่มต้นรันโปรแกรม สิ่งแรกที่จะแสดงบนหน้าจอคือหัวข้อว่าเป็นโปรแกรมที่ทำงานด้วยการเรียงลำดับแบบ Quick Sort และเส้นแบ่งที่ทำให้ข้อมูลมีการจัดระเบียบ



```
E:\ENGCE124\Coding 19 QUIC  x  +  v
ASCENDING QUICK SORT
=====
```

2. การแสดงข้อมูลดิบ (Raw Data)

หลังจากแสดงหัวข้อแล้ว โปรแกรมจะสุ่มตัวเลขจำนวน N (ในกรณีนี้ $N = 12$) และแสดงข้อมูลดิบที่ยังไม่ได้จัดเรียง โดยเรียกใช้ฟังก์ชัน `DispData()` ซึ่งจะแสดงข้อมูลตัวเลข 12 ตัว โดยข้อมูลดิบนี้จะถูกสุ่มขึ้นใหม่ทุกครั้งที่โปรแกรมทำงาน โดยค่าในอาร์เรย์จะเป็นตัวเลขตั้งแต่ 1 ถึง 99 และไม่เรียงลำดับ



```
Raw Data : 12 63 88 38 72 4 9 50 37 47 41 46
```

3. การแสดงขั้นตอนการจัดเรียง (Processing Data)

หลังจากแสดงข้อมูลดิบ โปรแกรมจะแสดงข้อความว่าโปรแกรมกำลังจัดเรียงข้อมูลอยู่ พร้อมกับแสดงลำดับตำแหน่งของข้อมูลในอาร์เรย์ จากตำแหน่งที่ 1 ถึงตำแหน่งที่ 12 เพื่อให้เห็นว่าการจัดเรียงจะเกิดขึ้นในแต่ละตำแหน่ง ในที่นี้ F และ R หมายถึงตำแหน่งของ pointer ที่ใช้ในการเปรียบเทียบและสลับค่า ซึ่งแสดงอยู่ในฟังก์ชัน `QuickSort()` โดย pointer F (front) และ R (rear) จะชี้ไปยังตำแหน่งต่าง ๆ ของข้อมูลในแต่ละรอบของการเปรียบเทียบ



```
Processing Data...
F R ( 1) ( 2) ( 3) ( 4) ( 5) ( 6) ( 7) ( 8) ( 9) (10) (11) (12)
```

4. การแสดงการเปลี่ยนแปลงระหว่างการจัดเรียง (Swapping and Sorting)

ในขั้นตอนนี้ โปรแกรมจะเริ่มกระบวนการจัดเรียงด้วย Quick Sort โดยจะมีการแสดงผลทุกครั้งที่มีการเปรียบเทียบและสลับตำแหน่งของข้อมูล เช่น หากโปรแกรมพบว่าข้อมูลที่ตำแหน่ง F (ตำแหน่งซ้าย) มีค่าสูงกว่าข้อมูลที่ตำแหน่ง R (ตำแหน่งขวา) โปรแกรมจะแสดงข้อมูลที่เปรียบเทียบกันอยู่ และทำการสลับ

ตำแหน่ง พร้อมแสดงข้อมูลที่เปลี่ยนแปลงแล้ว ข้อมูลที่แสดงในแต่ละบรรทัดจะบอกว่า pointer F และ R ชี้ไปที่ตำแหน่งใด และเมื่อทำการสลับค่าเสร็จแล้ว ผลลัพธ์ของอาเรย์จะเป็นอย่างไร กระบวนการนี้จะเกิดขึ้นซ้ำไปซ้ำมาจนกว่าข้อมูลทั้งหมดจะถูกจัดเรียงเรียบร้อยแล้ว

1	7	:	12	63	88	38	72	4	9	50	37	47	41	46
1	7	:	9	63	88	38	72	4	12	50	37	47	41	46
2	7	:	9	63	88	38	72	4	12	50	37	47	41	46
2	7	:	9	12	88	38	72	4	63	50	37	47	41	46
2	6	:	9	12	88	38	72	4	63	50	37	47	41	46
2	6	:	9	4	88	38	72	12	63	50	37	47	41	46
3	6	:	9	4	88	38	72	12	63	50	37	47	41	46
3	6	:	9	4	12	38	72	88	63	50	37	47	41	46

5. การแสดงค่าหลัก (Pivot) ในแต่ละรอบ

เมื่อ QuickSort() ทำการสลับค่าจน pointer F และ R ชี้ไปยังตำแหน่งเดียวกัน โปรแกรมจะแสดงค่าหลัก (pivot) ที่เลือกมาใช้ในการแบ่งข้อมูลออกเป็นสองส่วน คือส่วนที่น้อยกว่าค่าหลักและมากกว่าค่าหลัก ซึ่งค่า 12 ในที่นี้เป็นค่าหลัก (pivot) ซึ่งถูกใช้ในการแบ่งข้อมูลออกเป็นสองส่วน ก่อนที่โปรแกรมจะเรียก QuickSort() ซ้ำเพื่อจัดเรียงข้อมูลในแต่ละส่วน

```
k1=[12]-----
```

6. การแสดงผลข้อมูลที่เรียงลำดับแล้ว (Sorted Data)

หลังจากกระบวนการจัดเรียงทั้งหมดเสร็จสิ้น โปรแกรมจะแสดงเส้นแบ่งเพื่อบ่งบอกว่าการทำงานเสร็จสมบูรณ์แล้ว จากนั้นแสดงข้อมูลที่ถูกเรียงลำดับแล้วด้วยคำสั่ง DispData() ข้อมูลทั้งหมดที่ถูกสุ่มมาในตอนแรกจะถูกเรียงลำดับจากน้อยไปมากตามอัลกอริทึม Quick Sort

```
Sorted Data : 4 9 12 37 38 41 46 47 50 63 72 88
```

7. รอการกดปุ่มเพื่อสิ้นสุดโปรแกรม

หลังจากแสดงผลข้อมูลที่เรียงลำดับแล้ว โปรแกรมจะรอให้ผู้ใช้กดปุ่มใด ๆ ก่อนที่หน้าจอจะแสดงผลจะปิดลง ซึ่งทำได้โดยการใช้คำสั่ง getch()

```
-----
Sorted Data : 4 9 12 37 38 41 46 47 50 63 72 88
-----
Process exited after 786.1 seconds with return value 0
Press any key to continue . . . |
```

ผลลัพธ์ทั้งหมดของการทำงาน โปรแกรม Quick Sort

```

E:\ENGCE124\Coding 19 QUICKI × + v
ASCENDING QUICK SORT
=====
Raw Data : 12 63 88 38 72 4 9 50 37 47 41 46
Processing Data...
F R ( 1) ( 2) ( 3) ( 4) ( 5) ( 6) ( 7) ( 8) ( 9) (10) (11) (12)
1 7 : 12 63 88 38 72 4 9 50 37 47 41 46
1 7 : 9 63 88 38 72 4 12 50 37 47 41 46
2 7 : 9 63 88 38 72 4 12 50 37 47 41 46
2 7 : 9 12 88 38 72 4 63 50 37 47 41 46
2 6 : 9 12 88 38 72 4 63 50 37 47 41 46
2 6 : 9 4 88 38 72 12 63 50 37 47 41 46
3 6 : 9 4 88 38 72 12 63 50 37 47 41 46
3 6 : 9 4 12 38 72 88 63 50 37 47 41 46
k1=[12]-----
1 2 : 9 4 12 38 72 88 63 50 37 47 41 46
1 2 : 4 9 12 38 72 88 63 50 37 47 41 46
k1=[ 9]-----
4 9 : 4 9 12 38 72 88 63 50 37 47 41 46
4 9 : 4 9 12 37 72 88 63 50 38 47 41 46
5 9 : 4 9 12 37 72 88 63 50 38 47 41 46
5 9 : 4 9 12 37 38 88 63 50 72 47 41 46
k1=[38]-----
6 12 : 4 9 12 37 38 88 63 50 72 47 41 46
6 12 : 4 9 12 37 38 46 63 50 72 47 41 88
k1=[88]-----
6 11 : 4 9 12 37 38 46 63 50 72 47 41 88
6 11 : 4 9 12 37 38 41 63 50 72 47 46 88
7 11 : 4 9 12 37 38 41 63 50 72 47 46 88
7 11 : 4 9 12 37 38 41 46 50 72 47 63 88
k1=[46]-----
8 10 : 4 9 12 37 38 41 46 50 72 47 63 88
8 10 : 4 9 12 37 38 41 46 47 72 50 63 88
9 10 : 4 9 12 37 38 41 46 47 72 50 63 88
9 10 : 4 9 12 37 38 41 46 47 50 72 63 88
k1=[50]-----
10 11 : 4 9 12 37 38 41 46 47 50 72 63 88
10 11 : 4 9 12 37 38 41 46 47 50 63 72 88
k1=[72]-----
Sorted Data : 4 9 12 37 38 41 46 47 50 63 72 88

-----
Process exited after 786.1 seconds with return value 0
Press any key to continue . . . |

```

บรรณานุกรม

ChatGPT. (-). Exploring the Quick Sort Algorithm in C Language. สืบค้น 22 กันยายน 2567,
จาก <https://chatgpt.com/>