



รายงาน

เรื่อง โปรแกรม BUBBLE 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 BUBBLE 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 BUBBLE SORT รวมถึงอธิบายหลักการทำงานของโปรแกรม BUBBLE SORT และอธิบายผลลัพธ์การใช้งานโปรแกรม BUBBLE SORT

ผู้จัดทำรายงานหวังว่า รายงานฉบับนี้จะเป็นประโยชน์กับผู้สนใจ หรือนักศึกษาทุกท่านที่กำลังหาศึกษาในหัวข้อของโปรแกรม BUBBLE SORT หากมีข้อเสนอแนะหรือข้อผิดพลาดประการใด ผู้จัดทำขอน้อมรับไว้ และขออภัยมา ณ ที่นี้

ผู้จัดทำ

นายพงษ์พันธุ์ เลาวพงศ์

วันที่ 22/09/2567

สารบัญ

	หน้า
คำนำ	ก
สารบัญ	๗
โค้ดของโปรแกรม BUBBLE SORT พร้อมคำอธิบาย	1
หลักการทำงานของโปรแกรม BUBBLE SORT	5
ผลลัพธ์การใช้งานโปรแกรม BUBBLE SORT	11
บรรณานุกรม	13

โค้ดของโปรแกรม BUBBLE 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 N;

void PrepareRawData(int N)

{

    int i;

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

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

        Data[i] = 1 + rand() % 99; //สุ่มตัวเลขที่แตกต่างกันตั้งแต่ 1 ถึง 99

}

void DispData(int N)

{

    int i;

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

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

    printf("\n");

}
```

โค้ดของโปรแกรม BUBBLE SORT พร้อมคำอธิบาย (ต่อ)

```

void BubbleSort(int N)
{
    int i, j, temp;

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

    printf(" i ");

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

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

    printf("\n");

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

    for(i=1; i<=N-1; i++) //วนซ้ำไปข้างหน้า
    {
        if(Data[i] > Data[i+1]) //หากตำแหน่งไม่ถูกต้อง

        {
            printf("%2d. ", i+1);

            DispData(N);

            j = i + 1; //วนซ้ำไปข้างหลัง

            while(Data[j] < Data[j-1]) //ขณะที่ยังมี bubble ค้างอยู่

            {

                temp = Data[j-1]; //สลับข้อมูล

                Data[j-1] = Data[j];

                Data[j] = temp;
            }
        }
    }
}

```

โค้ดของโปรแกรม BUBBLE SORT พร้อมคำอธิบาย (ต่อ)

```

        j--; //นับถอยหลัง j

        printf("%2d. ", i+1);

        DispData(N);

    } //สิ้นสุด while

} //สิ้นสุด if

} //สิ้นสุด for

} //สิ้นสุดฟังก์ชัน

int main()

{

    printf("ASCENDING BUBBLE SORT\n");

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

    N = 12;

    PrepareRawData(N);

    printf("Raw Data : ");

    DispData(N);

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

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

    BubbleSort(N);

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

    printf("Sorted Data : ");

    DispData(N); //ข้อมูลที่ถูกจัดเรียง

```

โค้ดของโปรแกรม BUBBLE SORT พร้อมคำอธิบาย (ต่อ)

```
    getch();  
  
    return(0);  
} //สิ้นสุด main
```


หลักการการทำงานของโปรแกรม BUBBLE SORT

โปรแกรม BUBBLE SORT ใช้เทคนิค Ascending Bubble Sort นี้มีการทำงานหลักในการสลับข้อมูลตัวเลข จัดเรียงข้อมูลให้เป็นลำดับจากน้อยไปหามาก และแสดงข้อมูลทั้งก่อนและหลังการจัดเรียง

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 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

4. ฟังก์ชัน `PrepareRawData`

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

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

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

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

ฟังก์ชัน `PrepareRawData` ทำหน้าที่ในการเตรียมข้อมูลเริ่มต้น โดยสุ่มตัวเลขจำนวนเต็มในช่วง 1 ถึง 99 และเก็บไว้ใน Array `Data[]` ที่มีขนาด `N` ตัว โดยฟังก์ชันนี้ใช้ไลบรารี `stdlib.h` และ `time.h` ในการสุ่มตัวเลข และสร้างผลลัพธ์ที่แตกต่างกันทุกครั้งที่เราเรียกใช้ฟังก์ชัน โดยหลักการทำงานเริ่มต้นด้วยการเรียกใช้คำสั่ง `srand(time(NULL))` ซึ่งเป็นการกำหนดค่า `seed` ให้กับฟังก์ชัน `rand()` เพื่อให้การสุ่มตัวเลขมีความหลากหลายทุกครั้งที่เราเรียกใช้โปรแกรม จากนั้นใช้ลูป `for` เพื่อวนรอบตั้งแต่ `i=1` ถึง `i=N` โดยในแต่ละรอบจะใช้

rand() สุ่มตัวเลข และนำผลลัพธ์ที่ได้บวกกับ 1 เพื่อให้ค่าที่สุ่มได้อยู่ในช่วง 1 ถึง 99 (เพราะ rand() % 99 จะให้ค่าที่น้อยกว่า 99) สุดท้ายตัวเลขสุ่มแต่ละตัวจะถูกเก็บใน Array Data[]

5. ฟังก์ชัน DispData

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

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

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

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

ฟังก์ชัน DispData ทำหน้าที่แสดงข้อมูลใน Array Data[] โดยแสดงตัวเลขที่เก็บใน Array ออกมาเป็นแถวเดียวกัน พร้อมจัดรูปแบบให้มีช่องว่างระหว่างตัวเลข โดยหลักการทำงานเริ่มต้นจาก ใช้ลูป for เพื่อวนรอบแสดงตัวเลขที่เก็บใน Array ตั้งแต่ตำแหน่งที่ 1 ถึงตำแหน่งที่ N โดยในแต่ละรอบ จะใช้ printf("%2d ", Data[i]) เพื่อแสดงตัวเลขในตำแหน่ง i ของ Array โดยใช้ %2d เพื่อจัดรูปแบบให้ตัวเลขมีความกว้าง 2 หลัก และเมื่อแสดงครบทุกตัวแล้ว จะแสดง \n เพื่อขึ้นบรรทัดใหม่

6. ฟังก์ชัน BubbleSort

```
void BubbleSort(int N)
{
    int i, j, temp;

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

    printf(" i ");

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

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

6. ฟังก์ชัน BubbleSort (ต่อ)

```

printf("\n");

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

for(i=1; i<=N-1; i++) //วนซ้ำไปข้างหน้า
{
    if(Data[i] > Data[i+1]) //หากตำแหน่งไม่ถูกต้อง
    {
        printf("%2d. ", i+1);

        DispData(N);

        j = i + 1; //วนซ้ำไปข้างหลัง

        while(Data[j] < Data[j-1]) //ขณะที่ยังมี bubble ค้างอยู่
        {
            temp = Data[j-1]; //สลับข้อมูล

            Data[j-1] = Data[j];

            Data[j] = temp;

            j--; //นับถอยหลัง j

            printf("%2d. ", i+1);

            DispData(N);

        } //สิ้นสุด while

    } //สิ้นสุด if

} //สิ้นสุด for

} //สิ้นสุดฟังก์ชัน

```

6. ฟังก์ชัน BubbleSort (ต่อ)

ฟังก์ชัน BubbleSort ทำหน้าที่ในการจัดเรียงข้อมูลแบบ Bubble Sort โดยเป็นการจัดเรียงตัวเลขจากน้อยไปหามาก พร้อมกับแสดงรายละเอียดของแต่ละขั้นตอนในการจัดเรียง โดยหลักการทำงานเริ่มต้นด้วยการแสดงข้อมูลหัวข้อ โดยใช้เครื่องหมายขีด (-) เพื่อแบ่งการแสดงผลออกจากส่วนอื่น จากนั้นจะมีการแสดงตัวเลขดัชนี (i) ของแต่ละตำแหน่งใน Array เพื่อให้ผู้ใช้สามารถติดตามว่าข้อมูลถูกเปลี่ยนแปลงที่ตำแหน่งใด โดยโปรแกรมจะใช้ลูป for โดยวนรอบจาก $i=1$ ถึง $i=N-1$ เพื่อจัดเรียงข้อมูลในแต่ละรอบ ซึ่งภายในลูปจะมีการตรวจสอบเงื่อนไขว่า $Data[i] > Data[i+1]$ หากตัวเลขที่ตำแหน่ง i มีค่ามากกว่าตัวเลขที่ตำแหน่งถัดไป ($i+1$) หมายความว่าตัวเลขไม่เรียงกันถูกต้อง จึงต้องมีการสลับตำแหน่ง จากนั้น จะใช้ลูป while วนถอยหลัง ($j=i+1$) เพื่อตรวจสอบและสลับตัวเลขเรื่อย ๆ จนกว่าตัวเลขจะเรียงกันถูกต้อง ในทุกครั้งที่สลับตัวเลขเสร็จ ฟังก์ชัน DispData(N) จะถูกเรียกใช้เพื่อแสดงผลลัพธ์ของแต่ละขั้นตอนให้ผู้ใช้เห็น สุดท้ายกระบวนการนี้จะทำซ้ำไปเรื่อย ๆ จนกว่าตัวเลขทั้งหมดจะถูกจัดเรียงอย่างถูกต้อง

7. ฟังก์ชัน main

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

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

    N = 12;

    PrepareRawData(N);

    printf("Raw Data : ");

    DispData(N);

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

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

    BubbleSort(N);

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

7. ฟังก์ชัน main (ต่อ)

```
printf("Sorted Data : ");  
  
DispData(N); //ข้อมูลที่ถูกจัดเรียง  
  
getch();  
  
return(0);  
  
} //สิ้นสุด main
```

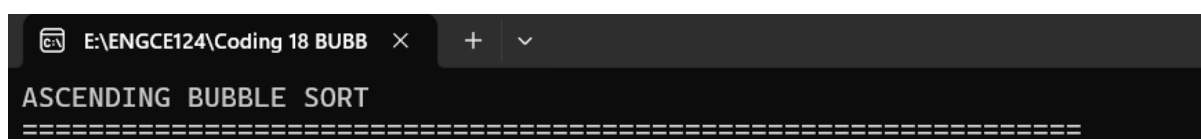
ฟังก์ชัน main() เป็นฟังก์ชันหลักของโปรแกรมที่เรียกใช้งานฟังก์ชันอื่น ๆ และควบคุมลำดับการทำงานทั้งหมดของโปรแกรม โดยหลักการทำงานเริ่มต้นด้วยการแสดงชื่อโปรแกรมและขีดเส้นแบ่งส่วน (=) จากนั้นกำหนดค่า $N = 12$ เพื่อใช้เป็นขนาดของ Array Data[] เมื่อกำหนดค่าแล้วจะเรียกฟังก์ชัน PrepareRawData(N) เพื่อสุ่มข้อมูลตัวเลขที่มีขนาด N ตัว และเรียกฟังก์ชัน DispData(N) เพื่อแสดงข้อมูลที่สุ่มได้ก่อนการจัดเรียง จากนั้นจะแสดงข้อความ "Processing Data..." เพื่อแจ้งผู้ใช้งานว่าข้อมูลกำลังถูกจัดเรียง เมื่อแสดงข้อความแล้วจะเรียกฟังก์ชัน BubbleSort(N) เพื่อจัดเรียงข้อมูลด้วยวิธี Bubble Sort เมื่อการจัดเรียงเสร็จสิ้น จะแสดงข้อมูลที่ถูกจัดเรียงแล้วโดยเรียกฟังก์ชัน DispData(N) อีกครั้ง สุดท้ายโปรแกรมจะหยุดรอการกดปุ่มจากผู้ใช้ด้วยคำสั่ง getch() และจากนั้นจะจบการทำงานของโปรแกรม

ผลลัพธ์การใช้งานโปรแกรม BUBBLE SORT

โปรแกรม Bubble Sort เป็นอัลกอริทึมการจัดเรียงข้อมูลแบบง่ายๆ ที่จัดเรียงข้อมูลจากน้อยไปมาก โดยทำการเปรียบเทียบและสลับตำแหน่งของตัวเลขในแต่ละขั้นตอน โปรแกรมจะเริ่มต้นด้วยการสุ่มตัวเลข แล้วจัดเรียงและแสดงผลการจัดเรียงในแต่ละขั้นตอนอย่างละเอียด

1. การเริ่มต้นการทำงานของโปรแกรม

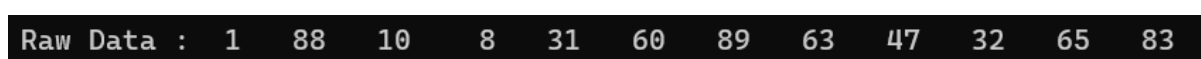
ทันทีที่โปรแกรมเริ่มทำงาน ระบบจะแสดงชื่อโปรแกรมและเส้นแบ่ง เพื่อให้ผู้ใช้ทราบว่าโปรแกรมนี้ทำงานเกี่ยวกับการจัดเรียงข้อมูลด้วย Bubble Sort โดยจะแสดงข้อความ "ASCENDING BUBBLE SORT" แสดงหัวข้อหลักของโปรแกรม พร้อมเส้นแบ่ง (=====) ที่ช่วยให้หน้าจอดูเป็นระเบียบและแยกส่วนต่าง ๆ ออกจากกันชัดเจน



```
E:\ENGCE124\Coding 18 BUBB  x  +  v
ASCENDING BUBBLE SORT
=====
```

2. การแสดงข้อมูลดิบที่เกิดจากการสุ่ม (Raw Data)

ต่อมา โปรแกรมจะสุ่มตัวเลขจำนวน 12 ตัว (ตามที่กำหนดในโค้ด) และแสดงข้อมูลดิบที่ยังไม่ได้ถูกจัดเรียงออกมาทางหน้าจอ ตัวเลขเหล่านี้ถูกสุ่มจากช่วง 1 ถึง 99 ซึ่งจะเป็นข้อมูลที่จะใช้ในการจัดเรียงด้วย Bubble Sort โดยตัวเลขที่สุ่มขึ้นมา จะเป็นข้อมูลดิบที่โปรแกรมจะทำการจัดเรียงในขั้นตอนถัดไป



```
Raw Data : 1 88 10 8 31 60 89 63 47 32 65 83
```

3. แสดงขั้นตอนการจัดเรียง (Bubble Sort)

เมื่อโปรแกรมเริ่มต้นการจัดเรียงข้อมูล จะแสดงรายละเอียดการเปรียบเทียบและสลับตำแหน่งของตัวเลขในแต่ละรอบ ผู้ใช้จะเห็นตัวเลขในแต่ละตำแหน่ง (i) ที่กำลังถูกพิจารณา และจะแสดงข้อมูลที่ถูกสลับหลังจากแต่ละการเปรียบเทียบ นอกจากนี้จะมีการแสดงขั้นตอนการจัดเรียงทุกครั้งที่เกิดการสลับตำแหน่ง เพื่อให้ผู้ใช้เห็นการเปลี่ยนแปลงของข้อมูล ในการแสดงผลลัพธ์ ตัวเลขที่ตำแหน่งที่ 1 (ค่า 23) มีค่ามากกว่าตัวเลขที่ตำแหน่งที่ 2 (ค่า 21) ดังนั้นโปรแกรมจะสลับตำแหน่งของทั้งสองตัวเลข หลังจากการสลับ โปรแกรมจะแสดงผลข้อมูลใหม่ให้ผู้ใช้งานเห็นทันที ซึ่งการแสดงผลตัวเลข 2. บนหน้าจอหมายถึงการตรวจสอบในรอบที่ 2 โดยแต่ละรอบจะตรวจสอบว่า ค่าปัจจุบันมีค่ามากกว่าค่าถัดไปหรือไม่ ถ้ามีจะเกิดการสลับตำแหน่ง และการแสดงผลแต่ละครั้งจะแสดงการเปลี่ยนแปลงของข้อมูลใน Array หลังจากที่มีการสลับตัวเลข ในการแสดงผลข้อมูลฟังก์ชัน DispData(N) ถูกเรียกใช้อย่างต่อเนื่องเพื่อแสดงสถานะของข้อมูลในแต่ละรอบการเปรียบเทียบ เมื่อการเปรียบเทียบและสลับตำแหน่งในรอบนั้นเสร็จสิ้น โปรแกรมจะวนซ้ำกลับไปตรวจสอบข้อมูลถัดไป

Processing Data...												
i	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
2.	23	21	22	26	70	6	22	57	97	90	62	65
2.	21	23	22	26	70	6	22	57	97	90	62	65
3.	21	23	22	26	70	6	22	57	97	90	62	65
3.	21	22	23	26	70	6	22	57	97	90	62	65
6.	21	22	23	26	70	6	22	57	97	90	62	65
6.	21	22	23	26	6	70	22	57	97	90	62	65
6.	21	22	23	6	26	70	22	57	97	90	62	65
6.	21	22	6	23	26	70	22	57	97	90	62	65
6.	21	6	22	23	26	70	22	57	97	90	62	65
6.	6	21	22	23	26	70	22	57	97	90	62	65
7.	6	21	22	23	26	70	22	57	97	90	62	65
7.	6	21	22	23	26	22	70	57	97	90	62	65
7.	6	21	22	23	22	26	70	57	97	90	62	65
7.	6	21	22	22	23	26	70	57	97	90	62	65
8.	6	21	22	22	23	26	70	57	97	90	62	65
8.	6	21	22	22	23	26	57	70	97	90	62	65
10.	6	21	22	22	23	26	57	70	97	90	62	65
10.	6	21	22	22	23	26	57	70	90	97	62	65
11.	6	21	22	22	23	26	57	70	90	97	62	65
11.	6	21	22	22	23	26	57	70	90	62	97	65
11.	6	21	22	22	23	26	57	70	62	90	97	65
11.	6	21	22	22	23	26	57	62	70	90	97	65
12.	6	21	22	22	23	26	57	62	70	90	97	65
12.	6	21	22	22	23	26	57	62	70	90	65	97
12.	6	21	22	22	23	26	57	62	70	65	90	97
12.	6	21	22	22	23	26	57	62	65	70	90	97

4. แสดงผลข้อมูลที่จัดเรียงเสร็จแล้ว

หลังจากโปรแกรมทำการจัดเรียงข้อมูลจนครบทุกขั้นตอน ข้อมูลทั้งหมดใน Array จะถูกเรียงจากน้อยไปมากแล้ว โปรแกรมจะแสดงผลข้อมูลที่จัดเรียงเรียบร้อยแล้วในส่วนท้ายของผลลัพธ์ สุดท้ายข้อมูลที่จัดเรียงเสร็จสมบูรณ์ในที่นี้คือ 6, 21, 22, 22, 23, 26, 57, 62, 65, 70, 90, และ 97 ซึ่งแสดงว่าการจัดเรียงข้อมูลด้วยเทคนิค Bubble Sort สำเร็จเรียบร้อยแล้ว

```
Sorted Data : 6 21 22 22 23 26 57 62 65 70 90 97
```

5. การรอการกดปุ่มก่อนปิดโปรแกรม

โปรแกรมจะไม่ปิดตัวลงทันทีหลังจากแสดงผลข้อมูลที่จัดเรียงเสร็จแล้ว มันจะรอจนกว่าผู้ใช้จะกดปุ่มใดๆ จากคีย์บอร์ด ก่อนที่โปรแกรมจะปิดการทำงาน ฟังก์ชัน getch() ทำหน้าที่นี้ เพื่อให้ผู้ใช้มีเวลาสังเกตผลลัพธ์ สุดท้ายก่อนออกจากโปรแกรม

```
Sorted Data : 6 21 22 22 23 26 57 62 65 70 90 97

Process exited after 389.4 seconds with return value 0
Press any key to continue . . . |
```


บรรณานุกรม

ChatGPT. (-). Bubble Sort: Step-by-Step Explanation with Output. สืบค้น 22 กันยายน 2567,
จาก <https://chatgpt.com/>