รายงาน : Lab03 – Array & Pointers

จัดทำโดย

นายกิตติพศ หนูทอง รหัสนักศึกษา : 6135512003

นายปฏิภาณ วรรณโก รหัสนักศึกษา : 6135512059

Section : 01

240-207 Programing and Data Structures

“งานทั้งหมดนี้ในรายงานฉบับนี้ล้วนเป็นผลงานของข้าพเจ้า มิได้ลอกหรือสำเนามาจากที่อื่นใด ในกรณีที่พบว่าเกิดสำเนาด้วยวิธีใดก็ตาม ข้าพเจ้ายินดีไม่ขอรับคะแนนจากรายงานฉบับนี้”

คะแนนที่ได้

ลงชื่อ

กิตติพศ หนูทอง ปฏิภาณ วรรณโก

(นายกิตติพศ หนูทอง) (นายปฏิภาณ วรรณโก)

Lab03 – Array & Pointers

ข้อที่ 1 : Sum with Pointer

Sum with Pointer

รับตัวเลขจำนวนเต็ม 5 ตัว แล้วหาผลบวกของสมาชิกทุกตัว ผ่านฟังก์ชัน `sum`

```
int sum(int *p, int *q);
```

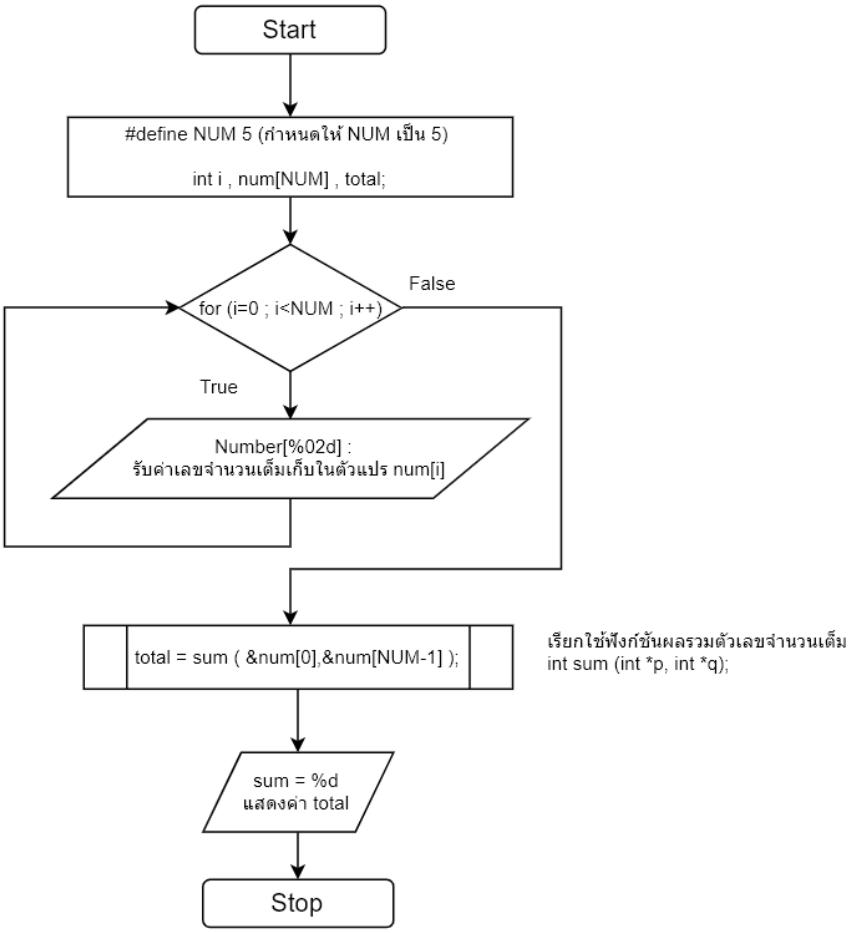
โดยห้ามใช้เครื่องหมาย [] ในฟังก์ชัน `sum` แต่ให้ใช้ประโยชน์จากการเพิ่มค่าของพอยเตอร์

Code

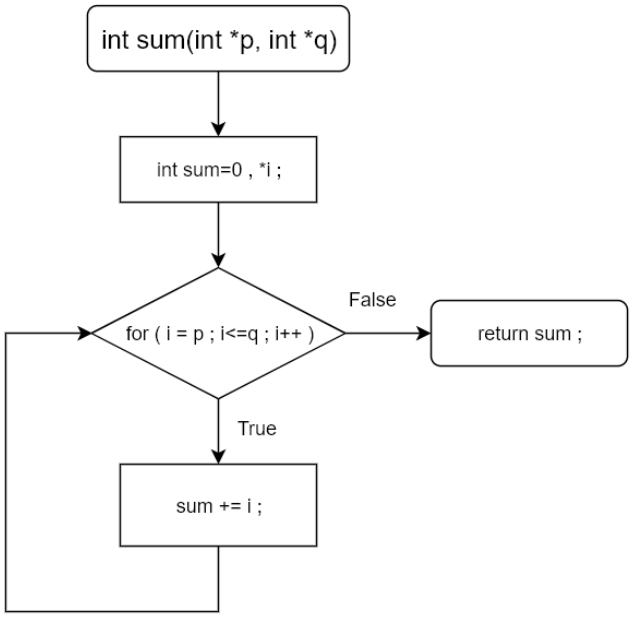
lab3-1.c

```c
#include<stdio.h>
#define NUM 5
int sum (int *p, int *q);
int main ()
{
    int i,num[NUM],total;
    for (i=0;i<NUM;i++)
    {
        printf("Number[%02d] : ",i+1);
        scanf("%d",&num[i]);
    }
    total = sum(&num[0],&num[NUM-1]);
    printf("sum = %d",total);
    return 0;
}
int sum(int *p, int *q)
{
    int sum=0,*i;
    for (i = p;i<=q;i++)
    {
        sum += *i;
    }
    return sum;
}
```

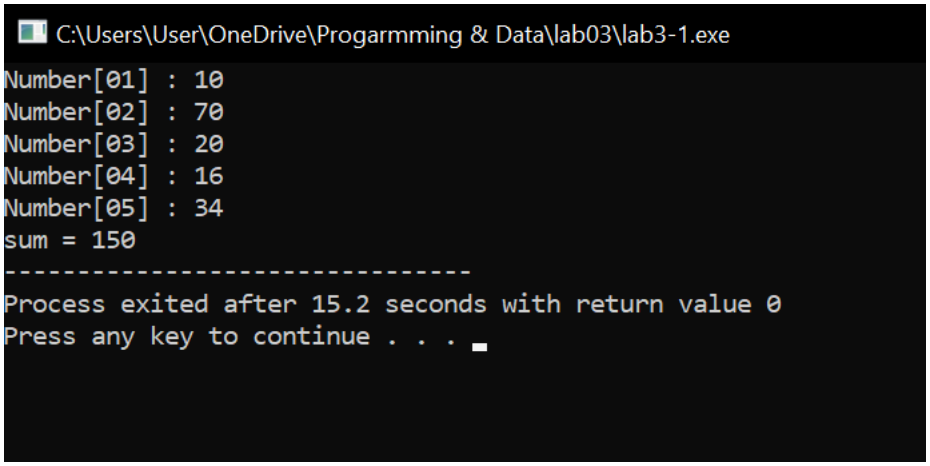
Flowchart



ฟังก์ชัน int sum(int *p, int *q)



ผลการรันโปรแกรม



อธิบายหลักการทำงาน : เป็นโปรแกรมที่รับค่าจำนวนเต็ม 5 ตัวแล้วหาผลบวกแสดงค่าออกมาทางหน้าจอ มีหลักการทำงานมีดังนี้ ในฟังก์ชัน main () จะประกาศตัวแปร i ไว้วนลูป , ตัวแปรอาร์เรย์ num[NUM] ซึ่งกำหนดค่า NUM คือ 5 (#define NUM 5) และตัวแปร total ไว้รับค่ารีเทิร์นกลับจากฟังก์ชันผลรวมตัวเลขจำนวนเต็ม int sum (int *p, int *q); โดยเริ่มแรกจะวนลูปรับค่าเลขจำนวนเต็มเก็บไว้ในตัวแปรอาร์เรย์ num[i] ภายใต้เงื่อนไขลูป for (i=0 ; i<NUM ; i++) หลังจากนั้นเรียกใช้และส่งค่า address ของตัวแปร num[0] ตำแหน่งแรก และ address ของตัวแปร num[NUM-1] ตำแหน่งสุดท้ายไปในฟังก์ชันผลรวมตัวเลขจำนวนเต็ม total = sum(&num[0],&num[NUM-1]); เก็บค่าไว้ในตัวแปร total แล้วแสดงค่า total ออกมาทางหน้าจอ

ในฟังก์ชัน int sum(int *p, int *q) จะประกาศตัวแปร sum = 0 และพอยน์เตอร์ integer *i โดยวนลูปให้พอยน์เตอร์ i = p และเลื่อนตำแหน่งพอยน์เตอร์ไปเรื่อยๆจนถึงพอยน์เตอร์ q ภายใต้เงื่อนไข for (i = p ; i<=q ; i++) หลังจากนั้นให้เก็บผลรวมไว้ในตัวแปร sum โดยค่า sum จะบวกกับค่าพอยน์เตอร์ *i ซึ่งพอยน์เตอร์ i จะชี้ไปยัง address ของตัวแปรอาร์เรย์ num ที่เก็บค่าจำนวนเต็มไว้ การบวกผลรวมของเลขจำนวนเต็มจะใช้คำสั่ง sum += *i; ต่อจากนั้นรีเทิร์นค่าตัวแปร sum กลับไปในฟังก์ชัน main ()

ความรู้จากการทำ lab : เมื่อมีการเปลี่ยนค่าของ pointer ที่ชี้ไปยัง address ตัวแปร จะทำให้ตัวแปรนั้นมีค่าที่เปลี่ยนไปด้วย

ข้อที่ 2 : Max to Sort

Max to Sort

รับค่าจำนวนเต็ม จำนวน 5 ตัว เก็บไว้ในอาเรย์ หลังจากนั้นใช้ประโยชน์จากฟังก์ชัน `find_max` เพื่อพิมพ์ข้อมูลเรียงจากมากไปหาน้อย โดยมีสมมุติฐานดังต่อไปนี้

1) ข้อมูลที่อยู่ในอาเรย์มากกว่าหรือเท่ากับ 0
2) สามารถเปลี่ยนแปลงข้อมูลในอาเรย์ได้

ทั้งนี้ฟังก์ชัน `find_max` มีต้นแบบของฟังก์ชันดังนี้

```
int* find_max(int *p, int *q);
```

=> `p` จุดเริ่มต้นของอาเรย์

=> `q` จุดสิ้นสุดของอาเรย์

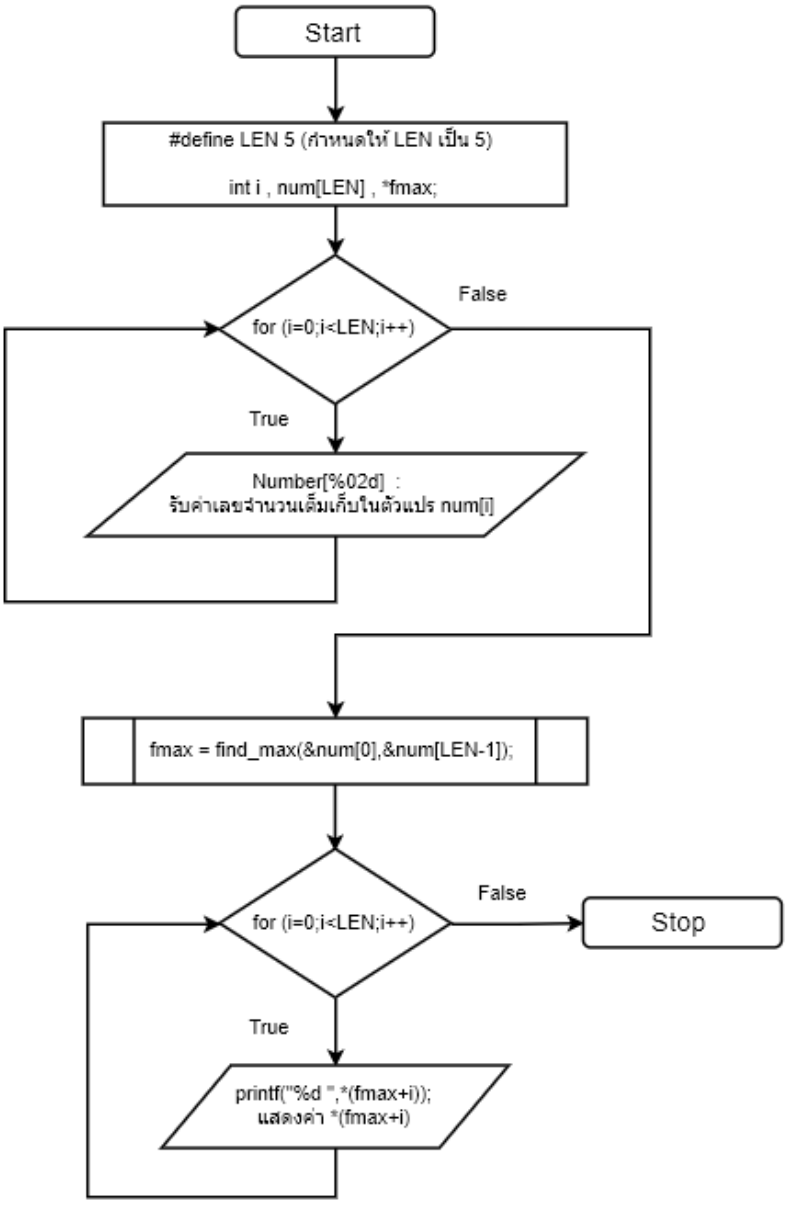
=> คืนค่า เป็น พอยเตอร์อ้างอิงไปยังสมาชิกที่มีค่ามากที่สุด

Code

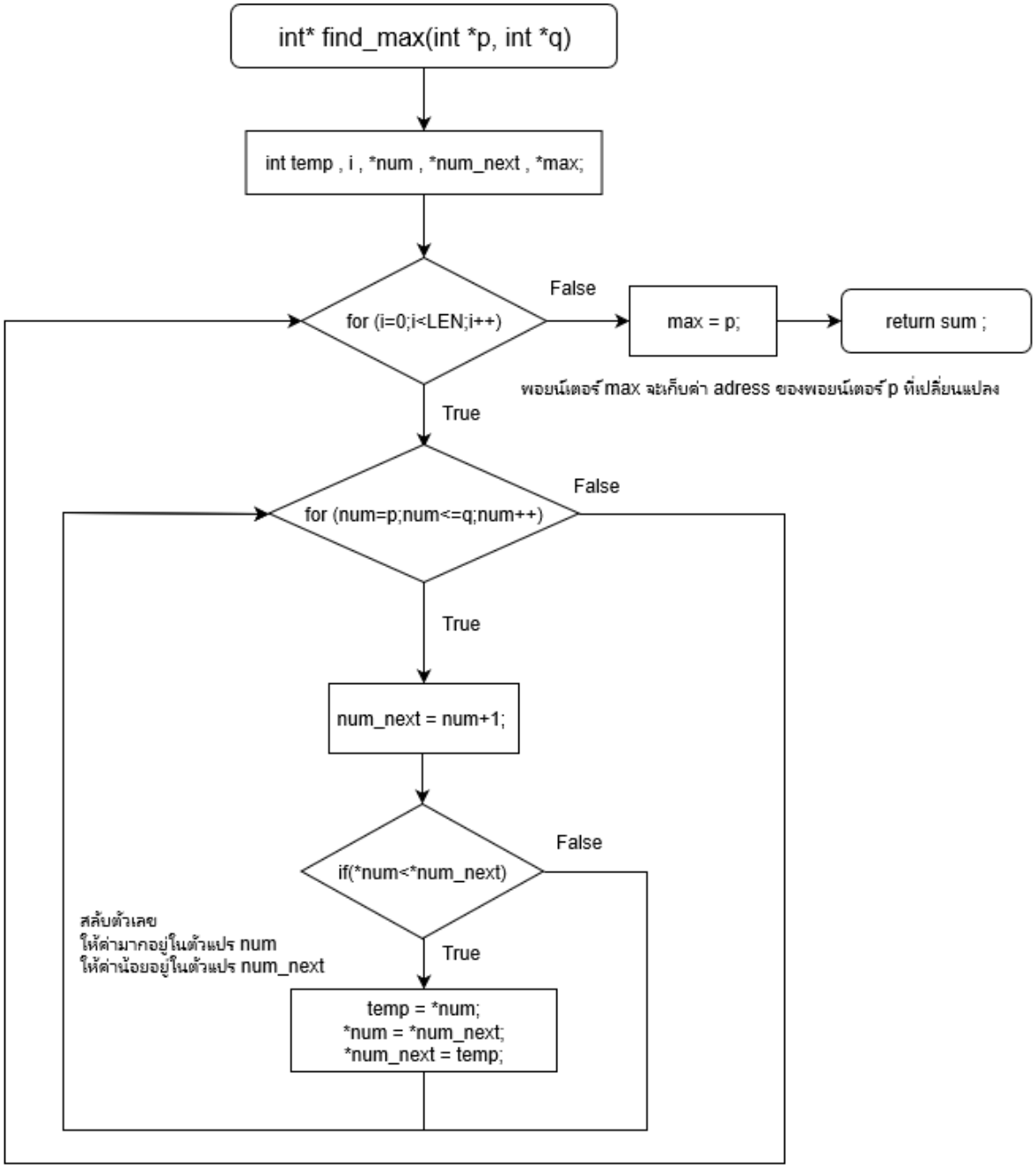
lab3-2.c

```c
#include<stdio.h>
#define LEN 5
int* find_max(int *p, int *q);
int main ()
{
    int i,num[LEN],*fmax;
    for (i=0;i<LEN;i++)
    {
        printf("Number[%02d] : ",i+1);
        scanf("%d",&num[i]);
    }
    fmax = find_max(&num[0],&num[LEN-1]);
    for (i=0;i<LEN;i++)
    {
        printf("%d ",*(fmax+i));
    }
    return 0;
}
int* find_max(int *p, int *q)
{
    int temp,i,*num,*num_next,*max;
    for (i=0;i<LEN;i++)
    {
        for (num=p;num<=q;num++)
        {
            num_next = num+1;
            if(*num<*num_next)
            {
                temp = *num;
                *num = *num_next;
                *num_next = temp;

            }

        }
    }
    max = p;
    return max;
}
```

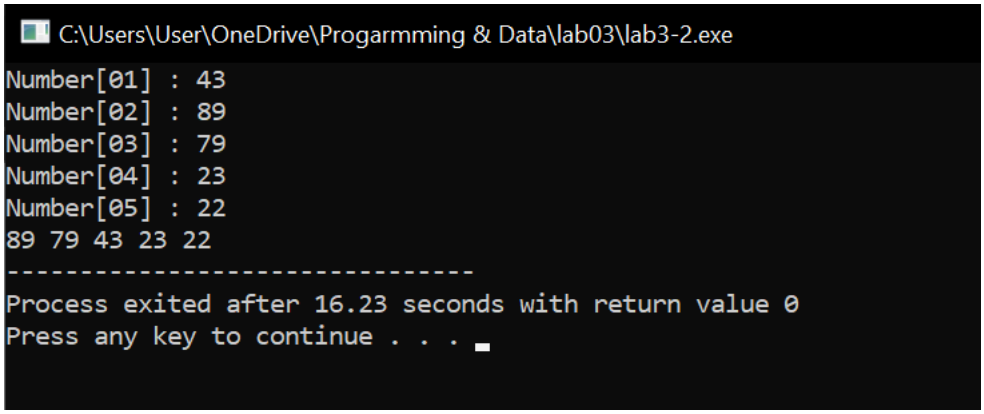
Flowchart


Start
#define LEN 5 (กำหนดให้ LEN เป็น 5)
int i , num[LEN] , *fmax;
for (i=0;i<LEN;i++)
False
True
Number[%02d] :
รับค่าเลขจำนวนเต็มเก็บในตัวแปร num[i]
fmax = find_max(&num[0],&num[LEN-1]);
for (i=0;i<LEN;i++)
False
Stop
True
printf("%d ",*(fmax+i));
แสดงค่า *(fmax+i)

ฟังก์ชัน int* find_max(int *p, int *q)



ผลการรันโปรแกรม



อธิบายหลักการทำงาน : โปรแกรมนี้เป็นการรับค่าของจำนวนเต็ม 5 ตัวมาเพื่อค้นหาจำนวนที่มากที่สุดและเรียงลำดับจากมากไปหาน้อยที่สุดและนำมาแสดงค่าบนหน้าจอ จาก code จะแสดงให้เห็นถึงการแยกฟังก์ชันสองตัวคือ ฟังก์ชัน main และฟังก์ชัน find_max โดยที่ในฟังก์ชัน main จะมีการประกาศตัวแปรประเภท int สองตัวได้แก่ i (นำมาวนลูป) num[LEN] (เป็นอาร์เรย์เก็บค่าจำนวนเต็ม) ซึ่ง LEN คือการประกาศค่าก่อนนำมาเข้าฟังก์ชันซึ่ง LEN เท่ากับ 5 และค่าที่ Pointer หนึ่งตัวคือ *fmax และนำไปวนลูปโดยใช้ ตัวแปร i นำมาวนลูบโดยให้เงื่อนไข for (i = 0 ; i < LEN ; i++) จากนั้นจะเห็นการใช้ printf เพื่อแสดงข้อความ Number[%02d] : เพื่อให้ผู้ใช้รู้ว่าเป็นจำนวนที่เท่าไรและให้ใส่ค่าใด จากนั้นจะเป็นการใช้ scanf เพื่อรับค่าจำนวนเต็มจากผู้ใช้ หลังจากนั้น วนลูปจนครบตัวเงื่อนไขและออกจากลูบ ตามด้วยการวนลูบการเรียกใช้ฟังก์ชั่น find_max โดยที่ส่งค่า address ของ num[0] และ num[LEN-1] แล้วนำค่ามาเก็บในตัวแปร fmaxและใช้ลูปเงื่อนไข for (i = 0 ; i < LEN ; i++) เพื่อ printf แสดงค่าจำนวนเต็มของตัวแปร fmax โดยใช้ค่าของตัวแปร *(fmax+i) ซึ่งค่าของตำแหน่ง fmax จะเลื่อนไปเรื่อย ๆ จนกว่าจะหมดเงื่อนไข

ฟังก์ชัน find_max จะมีการประกาศตัวแปรประเภท int 2 ตัว ตัวแปร temp ใช้ในการสลับตัวเลข และตัวแปร i ใช้ในการวนลูป และค่าที่ Pointer สองตัว num คือการรับค่า address ของจำนวนเต็มของอาเรย์ที่ส่งมาเพื่อวนลูป ส่วน num_next คือค่าจำนวนอาร์เรย์ถัดไปจาก num และ max จะรับค่า address ที่ทำการเปลี่ยนแปลงเรียบร้อย และตั้งเงื่อนไข for (i = 0 ; i < LEN ; i++) เพื่อวนลูปตามจำนวนของค่าจำนวนเต็มที่ป้อนเข้ามา และลูปอีกชั้นมีเงื่อนไข for (num=p ; num<=q ; num++) เพื่อสลับค่าของเลขจำนวนเต็ม โดยนำค่ามาเปรียบเทียบและประกาศให้ค่าของ num_next = num+1; เปรียบเทียบค่า *num กับ *num_next โดยมีเงื่อนไข if (*num < *num_next) ถ้าตรงเงื่อนไขให้ทำดังต่อไปนี้

ค่าของ temp จะเก็บค่าที่ num ชี้ (temp = *num;)

ค่าที่ num ชี้ จะเก็บค่าที่ num_next ชี้ (*num = *num_next;)

ค่าที่ num_next ชี้ จะเก็บค่าของ temp (*num_next = temp;)

จากนั้นวนลูปจนครบเงื่อนไขและออกจากลูบ ประกาศให้ค่า max = p; เพื่อรับค่า address ของ p ที่เปลี่ยนแปลงและส่งค่า max กลับฟังก์ชั่น main

ความรู้จากการทำ lab : การได้ใช้ for loop ในการสลับตัวเลขหรือการทำ bubble sort และการส่งค่าและคืนค่าของ address และ pointer ที่ถูกต้อง

ข้อที่ 3 : Another Find Max

Another Find Max

รับค่าจำนวนเต็ม จำนวน 5 ตัว เก็บไว้ในอาเรย์ หลังจากนั้นใช้ประโยชน์จากฟังก์ชัน `max` เพื่อหาค่ามากที่สุด
ทั้งนี้ฟังก์ชัน `max` มีต้นแบบของฟังก์ชันดังนี้

```
int* max(int *a, int *b);
```

=> `a` ตำแหน่งของตัวแปรตัวที่ 1

=> `b` ตำแหน่งของตัวแปรตัวที่ 2

=> คืนค่า เป็น พอยเตอร์อ้างอิงไปยังตัวแปรที่มีค่ามากกว่า

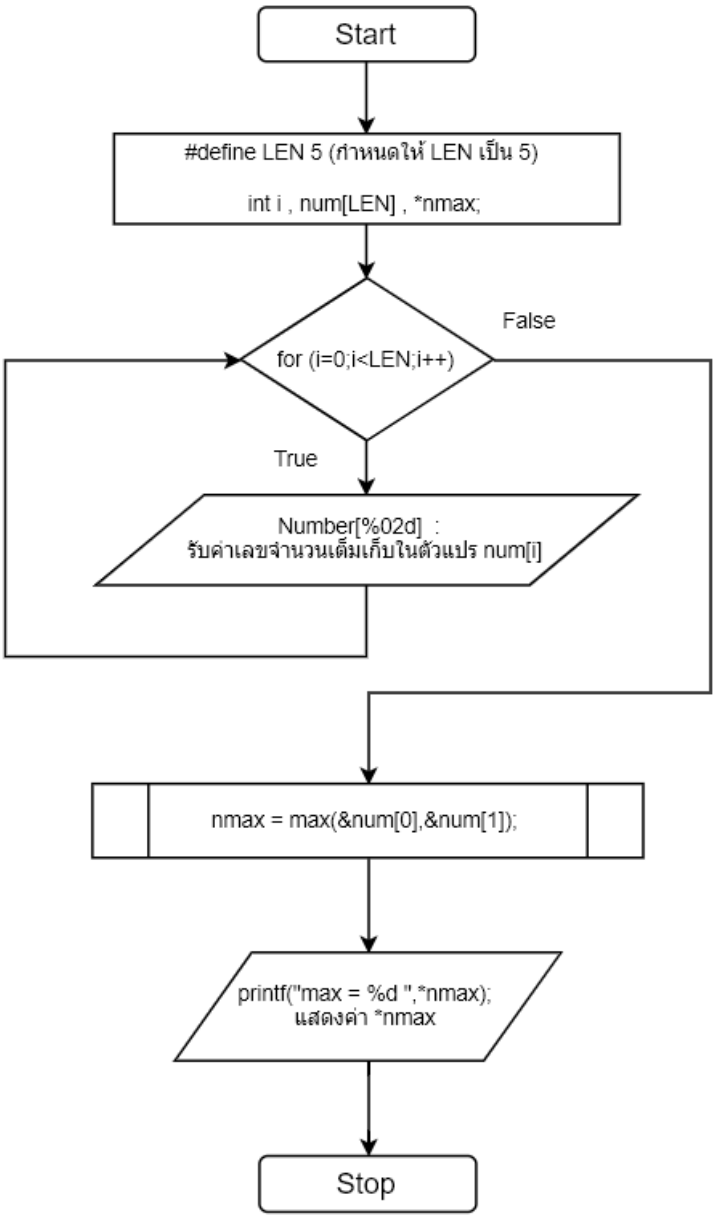
ข้อสังเกต ฟังก์ชันนี้ประมวลผลกับตัวแปรเดี่ยว ไม่ใช่อาเรย์

Code

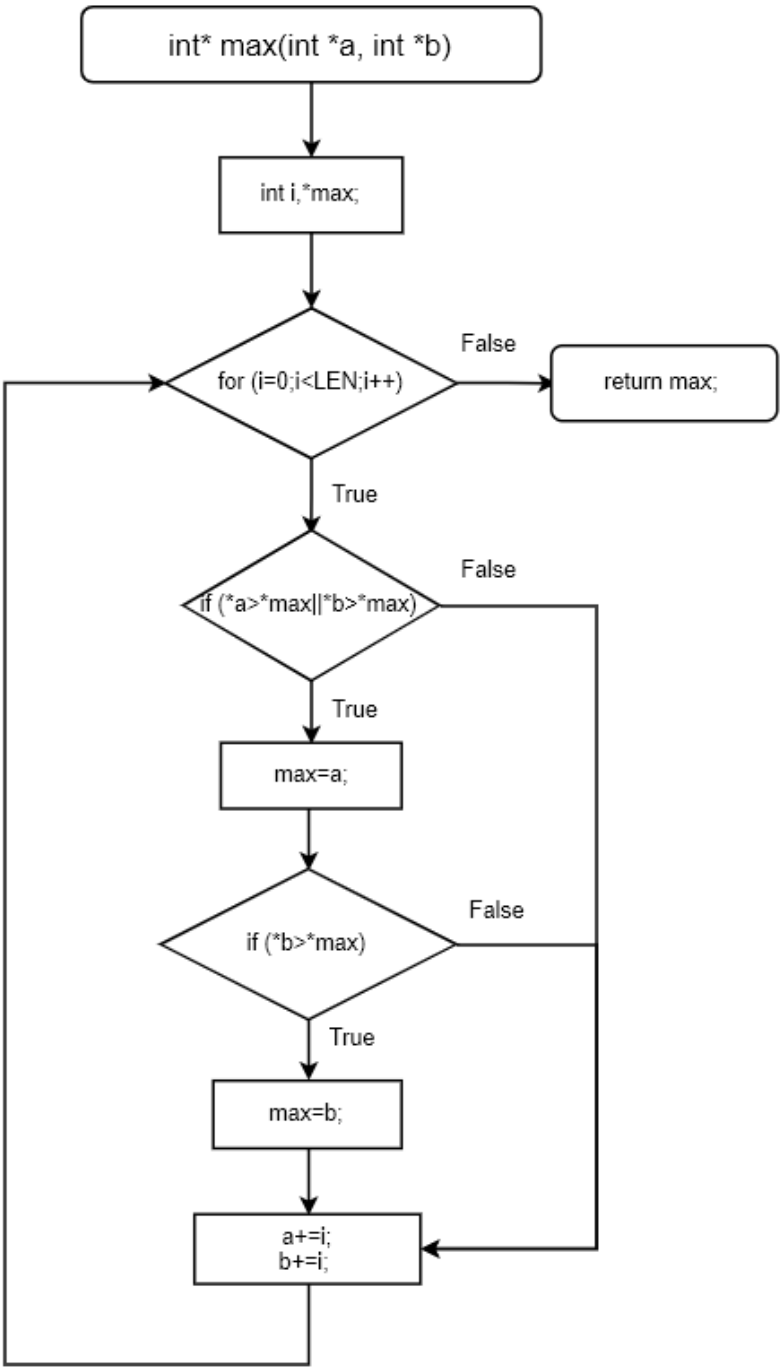
lab3-3.c

```c
#include<stdio.h>
#define LEN 5
int* max(int *a, int *b);
int main ()
{
    int i,num[LEN],*nmax;
    for (i=0;i<LEN;i++)
    {
        printf("Number[%02d] : ",i+1);
        scanf("%d",&num[i]);
    }
    nmax = max(&num[0],&num[1]);

    printf("max = %d ",*nmax);
    return 0;
}
int* max(int *a, int *b)
{
    int i,*max;
    max = a;
    for (i=0;i<LEN-1;i++)
    {
        if (*a>*max||*b>*max)
        {
            max=a;
            if (*b>*max)
            {
            max=b;
            }
        }
    a+=i;
    b+=i;
    }
    return max;
}
```

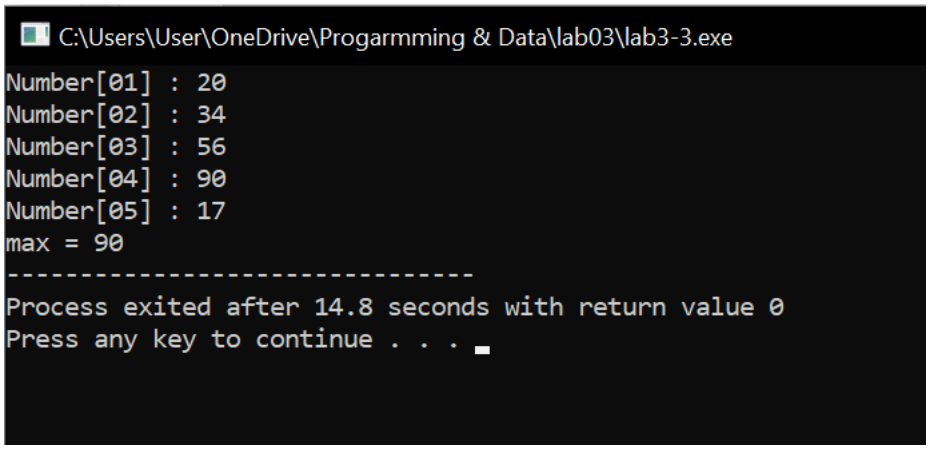
Flowchart


Start
#define LEN 5 (กำหนดให้ LEN เป็น 5)
int i , num[LEN] , *nmax;
for (i=0;i<LEN;i++)
False
True
Number[%02d] :
รับค่าเลขจำนวนเต็มเก็บในตัวแปร num[i]
nmax = max(&num[0],&num[1]);
printf("max = %d ",*nmax);
แสดงค่า *nmax
Stop

ฟังก์ชัน int* max(int *a, int *b)



ผลการรันโปรแกรม



อธิบายหลักการทำงาน : โปรแกรมนี้เป็นการรับค่าของจำนวนเต็ม 5 ตัวมาเพื่อค้นหาจำนวนที่มากที่สุดจาก code จะแสดงให้เห็นถึงการแยกฟังก์ชันสองตัวคือ ฟังก์ชั่น main และฟังก์ชั่น max โดยที่ในฟังก์ชั่น main จะมีการประกาศตัวแปรประเภท int สองตัวได้แก่ i (นำมาวนลูป) num[LEN] (เป็นอาร์เรย์เก็บค่าจำนวนเต็ม) ซึ่ง LEN คือการประกาศค่าก่อนนำมาเข้าฟังก์ชั่นโดยที่ LEN เท่ากับ 5 และค่าที่ Pointer หนึ่งตัวคือ *nmax และนำไปวนลูปโดยใช้ตัวแปร i นำมาวนลูปโดยให้เงื่อนไข for (i = 0 ; i < LEN ; i++) จากนั้นจะเห็นการใช้ printf เพื่อแสดงข้อความ Number[%02d] : เพื่อให้ผู้ใช้รู้ว่าเป็นจำนวนที่เท่าไรและให้ใส่ค่าใด จากนั้นจะเป็นการใช้ scanf เพื่อรับค่าจำนวนเต็มจากผู้ใช้เข้ามาในตัวแปร num ตัวที่ i หลังจากนั้นวนลูปจนครบเงื่อนไขและออกจากลูป เรียกใช้ฟังก์ชัน max โดยที่ส่งค่า address ของ num[0] และ num[1]และนำค่ามาเท่ากับค่าของ fmax และใช้printf เพื่อแสดงค่าของ nmax บนหน้าจอ

ฟังก์ชัน max จะมีการประกาศตัวแปรประเภท int 1 ตัว i เพื่อนำมาวนลูป และค่าที่ Pointer 1 ตัว คือ max เพื่อรับค่า address ของจำนวนเต็มของอาร์เรย์ที่มากที่สุดเพื่อส่งกลับ ต่อจากนั้นจะเป็นการประกาศให้ค่าของ max เท่ากับค่าของ a ซึ่งเป็นค่า address num[0] เพื่อกำหนดค่า max ไว้เปรียบเทียบ หลังจากนั้นให้เงื่อนไขเป็น for (i = 0 ; i < LEN ; i++) เพื่อวนหาค่าที่มากที่สุด และใช้ if else โดยมีเงื่อนไขว่า if (*a > *max || *b > *max) ถ้าตรงเงื่อนไขให้ค่า max เท่ากับค่า a และใช้ if else อีกรอบเพื่อเช็ดความผิดพลาดที่อาจเกิดได้ โดยมีเงื่อนไขว่า if (*b>*max) ถ้าใช่ให้ค่า max เท่ากับค่า b แต่ถ้าไม่ใช่ค่า max จะเท่ากับค่า a เช่นเดิม หลังจากนั้นให้ค่า a++; b++; วนลูปจนครบเงื่อนไข และคืนค่า max กลับฟังก์ชัน main

ความรู้จากการทำ lab : ในฟังก์ชัน array ไม่จำเป็นต้องส่ง address ไปทั้งหมดเพียงส่งไปไม่กี่ตำแหน่งได้ไปใน pointer และเลื่อนตำแหน่งหาค่าของ address ของ array ของตำแหน่งได้จากการเลื่อนตำแหน่งของ pointer