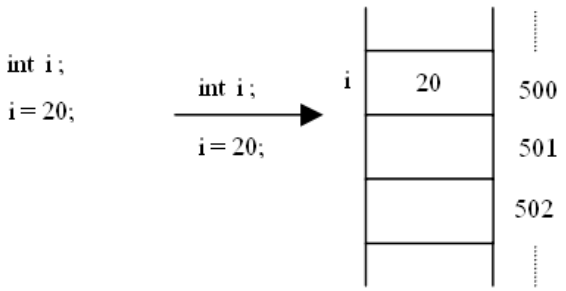
หน่วยเรียนที่ 7

พอยน์เตอร์

พอยน์เตอร์ (Pointer) เป็นชนิดข้อมูลชนิดหนึ่งในภาษาซี ซึ่งมีความเร็วในการทำงานสูง และช่วยให้การเขียนภาษาซีมีความยืดหยุ่น การใช้งานพอยน์เตอร์อาจมีความซับซ้อน แต่หากมีความเข้าใจแล้วจะทำให้การพัฒนาภาษาซีเป็นไปด้วยความสะดวกมากยิ่งขึ้น

7.1 พอยน์เตอร์กับตำแหน่งหน่วยความจำ

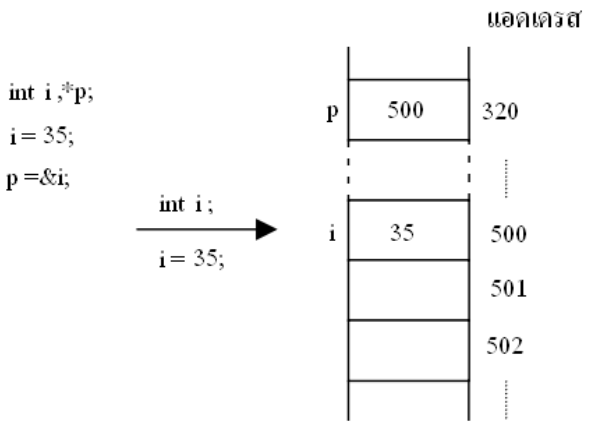
การประกาศตัวแปรที่ผ่านมา เช่น การประกาศ char i; นอกจากจะเป็นการตั้งชื่อตัวแปร i แล้วยังเป็นการจองหน่วยความจำสำหรับตัวแปร i อีกด้วย โดยตัวแปร i จะมีตำแหน่งความจำขนาด 8 บิตสำหรับเก็บค่าข้อมูล โดยลำดับตำแหน่งใดนั้น จะขึ้นกับคอมไพเลอร์เป็นตัวกำหนด ดังนั้นเมื่อมีการประกาศตัวแปร จะมีการจองพื้นที่หน่วยความจำที่มีขนาดเท่ากับขนาดหน่วยความจำข้อมูลชนิดนั้นๆ ซึ่งการเข้าถึงข้อมูลจะสามารถเข้าถึงโดยการระบุชื่อตัวแปรนั้นๆ หรืออาจจะระบุตำแหน่งของหน่วยความจำนั้นๆก็ได้ โดยการจองหน่วยความจำมีลักษณะดังในภาพที่ 7.1



ภาพที่ 7.1 การจองและการเข้าถึงข้อมูลหน่วยความจำ

สมมติว่ามีการจองหน่วยความจำที่ตำแหน่งแอดเดรส 500 ให้กับตัวแปร i ขนาด 1 ไบต์ เมื่อมีการนำตัวแปรนั้นมาใช้งาน ด้วยคำสั่งกำหนดค่า i = 20; จะเป็นการกำหนดให้ตัวแปร i มีค่าเท่ากับ 20 หรือกล่าวอีกอย่างว่าเป็นการกำหนดให้ตำแหน่งหน่วยความจำที่ 500 มีค่าเท่ากับ 20

ดังนั้นการเข้าถึงข้อมูลโดยระบุตำแหน่งหน่วยความจำจะเป็นลักษณะการใช้พอยน์เตอร์มาเป็นตัวระบุตำแหน่งหรือชี้ตำแหน่งหน่วยความจำ ดังนั้นตัวแปรพอยน์เตอร์เป็นข้อมูลชนิดหนึ่งที่แตกต่างจากข้อมูลพื้นฐานอื่นๆ เนื่องจากตัวแปรพอยน์เตอร์เป็นตัวแปรที่ใช้เก็บค่าตำแหน่งหน่วยความจำของตัวแปรอื่นๆ หากตัวแปร i เป็นตัวแปรประเภท char และตัวแปร p เป็นตัวแปรประเภทพอยน์เตอร์ที่เก็บตำแหน่งหน่วยความจำของตัวแปร i (ตัวแปร p ชี้ตำแหน่งตัวแปร i) ซึ่งสามารถจำลองลักษณะการจองตำแหน่งหน่วยความจำดังภาพที่ 7.2



ภาพที่ 7.2 การจองและการเข้าถึงข้อมูลหน่วยความจำ

สมมติให้ตัวแปร p ซึ่งเป็นตัวแปรพอยน์เตอร์อยู่ที่ตำแหน่งหน่วยความจำ 320 ตัวแปร i อยู่ที่ตำแหน่งหน่วยความจำ 500 และเก็บค่าข้อมูล 35 ดังนั้นเมื่อกำหนดให้ตัวแปร p เป็นพอยน์เตอร์ชี้ตำแหน่งหน่วยความจำของตัวแปร i จะทำให้ตัวแปร p มีค่าเท่ากับตำแหน่งหน่วยความจำของตัวแปร i คือ 500 ซึ่งเก็บค่า 35 อยู่ในตำแหน่งหน่วยความจำ

7.1.1 การประกาศตัวแปรพอยน์เตอร์

การประกาศตัวแปรพอยน์เตอร์จะใช้การดำเนินการชนิดเอกภาค (Unary Operator) หรือเครื่องหมาย'*' นำหน้าชื่อตัวแปรประเภทพอยน์เตอร์ โดยมีรูปแบบดังนี้

ชนิดข้อมูล * ชื่อตัวแปร ;

เช่น

```
int  *ip;       // ประกาศตัวแปร ip เป็นตัวแปรพอยน์เตอร์ชี้ไปยังตัวแปรชนิด int
float *dp;      // ประกาศตัวแปร dp เป็นตัวแปรพอยน์เตอร์ชี้ไปยังตัวแปรชนิด float
float atof(char *); //  ประกาศฟังก์ชัน atof มีพารามิเตอร์เป็นตัวแปรพอยน์เตอร์
                    //  ชนิด char และมีการคืนค่ากลับเป็น float
```

7.1.2 การกำหนดตำแหน่งให้แก่ตัวแปรพอยน์เตอร์

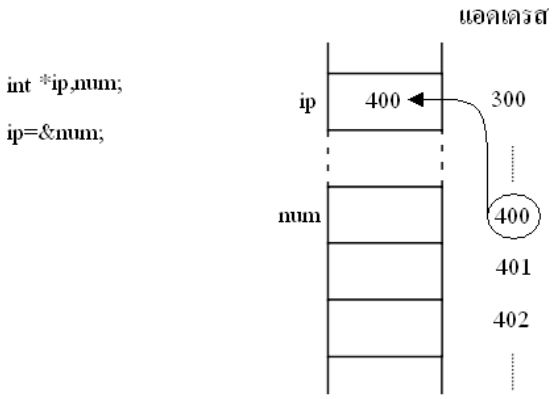
การกำหนดตำแหน่งหน่วยความจำให้แก่ตัวแปรพอยน์เตอร์ ซึ่งเป็นการกำหนดค่าตำแหน่งหน่วยความจำของตัวแปรที่มีข้อมูลสอดคล้องกับชนิดข้อมูลของตัวแปรพอยน์เตอร์ โดยการใช้การดำเนินการชนิดเอกภาค (Unary Operator) ซึ่งใช้เครื่องหมาย '&' นำหน้าตัวแปรที่ต้องการระบุตำแหน่งหน่วยความจำ โดยมีลักษณะการใช้งานดังนี้

ตัวแปรพอยน์เตอร์ = & ตัวแปร;

เช่น

```
int *ip, num;               //  ประกาศ ip เป็นตัวแปรพอยน์เตอร์
ip = &num;                  //  กำหนดให้ ip ชี้ไปยังตำแหน่งหน่วยความจำของ num
```

ซึ่งหมายถึงมีการประกาศตัวแปร num และตัวแปรพอยน์เตอร์ *ip เป็นชนิด int และบรรดทัดถัดมาเป็นการกำหนดตำแหน่งหน่วยความจำของตัวแปร num ให้แก่ตัวแปรพอยน์เตอร์ ip หรือหมายถึงตัวแปร ip เก็บตำแหน่งหรือชี้ไปยังตำแหน่งหน่วยความจำของตัวแปร num ดังในภาพที่ 7.3 นั่นคือสมมติให้ตัวแปร num อยู่ที่ตำแหน่งหน่วยความจำ 400 เมื่อมีการกำหนด ip = # จะหมายถึงกำหนดตัวแปร ip มีค่าเท่ากับให้ตำแหน่งหน่วยความตัวแปร num ซึ่งคือตำแหน่งหน่วยความจำลำดับที่ 400 นั่นเอง



ภาพที่ 7.3 การกำหนดตำแหน่งหน่วยความจำให้แก่ตัวแปรพอยน์เตอร์

7.1.3 การกำหนดค่าและอ่านค่าข้อมูลจากตัวแปรพอยน์เตอร์

การกำหนดค่าและอ่านค่าข้อมูลจากตัวแปรพอยน์เตอร์ เป็นการเข้าถึงค่าข้อมูลของตำแหน่งหน่วยความของตัวแปรใดๆ ที่ตัวแปรพอยน์เตอร์ชี้ตำแหน่งอยู่นั่นเอง ซึ่งหมายถึงเป็นการอ่านค่าหรือ อาจเป็นการกำหนดค่าให้แก่ตำแหน่งหน่วยความจำที่ตัวแปรพอยน์เตอร์ชี้ตำแหน่งอยู่นั่นเอง โดยรูปแบบการอ่านข้อมูลจากตัวแปรพอยน์เตอร์เป็นดังนี้

ตัวแปร = *ตัวแปรพอยน์เตอร์;

เช่น

```
int *ip,num1,num2;
num1 = 45;
ip =&num1;
num2 = *ip;
```

หมายถึง มีการกำหนดตัวแปร num1 num2 เป็นตัวแปรชนิด int และ ip เป็นตัวแปรพอยน์เตอร์ มีการกำหนดให้พอยน์เตอร์ ip ชี้ตำแหน่งหน่วยความจำของตัวแปร num1 ซึ่ง ณ ตำแหน่งหน่วยความจำตัวแปร num1 มีค่าข้อมูลเท่ากับ 45 จากนั้นกำหนดให้ num2 มีค่าเท่ากับข้อมูลในตำแหน่งที่พอยน์เตอร์ ip ชี้ตำแหน่งของ num1 ซึ่งก็เท่ากับ 45 นั่นเอง

ตัวอย่างที่ 7.1 โปรแกรมชี้ตำแหน่งหน่วยความจำด้วยพอยน์เตอร์

```
#include <stdio.h>
void main(void)
{
  int X,Y;
  int *ip;
  X = 19;
  ip = &X;                  // กำหนดพอยน์เตอร์ชี้ตำแหน่งหน่วยความจำ X
  Y = *ip;                  // อ่านข้อมูลของตำแหน่งที่พอยน์เตอร์ชี้อยู่มาเก็บใน Y
  printf(" The value of Y = %d",Y);
}
```

ในโปรแกรมมีการประกาศตัวแปร X และ Y เป็นตัวแปรชนิด int และประกาศตัวแปร ip เป็นตัวแปรพอยน์เตอร์ แล้วมีการกำหนดค่าให้ตัวแปร X มีค่าเท่ากับ 19 และกำหนดตัวแปร i มีค่าเท่ากับตำแหน่งหน่วยความของตัวแปร X จากนั้น ตัวแปร Y มีค่าเท่ากับค่าข้อมูลของตำแหน่งหน่วยความจำที่ตัวแปร ip ระบุไว้ ซึ่งคือค่าข้อมูล 45 นั่นเอง โดยผลการทำงานของโปรแกรมดังแสดงในภาพที่ 7.4

```
The value of  Y = 19
```

ภาพที่ 7.4 ผลการทำงานของโปรแกรมตัวอย่างที่ 7.1

7.2 พอยน์เตอร์ชี้ตำแหน่งตัวแปรอาเรย์

การกำหนดตัวแปรพอยน์เตอร์ชี้ตำแหน่งหน่วยความจำตัวแปรอาเรย์ จะมีความเกี่ยวข้องกันและมีการใช้งานกันมาก เนื่องจากอาเรย์เป็นตัวแปรที่เก็บข้อมูลชนิดเดียวกันเป็นชุดตัวแปร และตำแหน่งหน่วยความจำของตัวแปรมีการเรียงลำดับชิดติดกัน ทำให้การเพิ่มหรือลดค่าตัวชี้

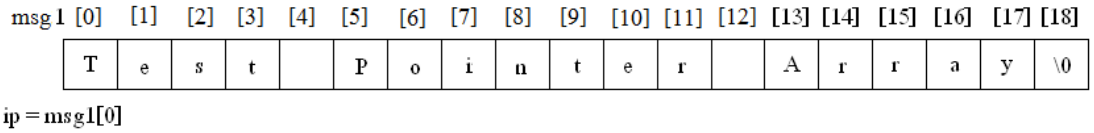
ตำแหน่งหน่วยความจำทำได้ง่าย ซึ่งการกำหนดตำแหน่งหน่วยความจำของตัวแปรอาเรย์ ให้แก่ตัวแปรพอยน์เตอร์ หากไม่ได้ระบุตำแหน่งสมาชิกภายใน ก็จะเป็นการกำหนดตำแหน่งสมาชิกตำแหน่ง [0] ให้แก่ตัวแปรพอยน์เตอร์โดยอัตโนมัติ โดยรูปแบบการกำหนดตำแหน่งหน่วยความจำของตัวแปรอาเรย์ให้ตัวแปรพอยน์เตอร์ชี้ตำแหน่งเป็นดังนี้

ตัวแปรพอยน์เตอร์ = ตัวแปรอาเรย์;

เช่น

```
int  msg1[20];          // ประกาศตัวแปรอาเรย์ msg1 จำนวน 20 สมาชิก
int *ip;                // ประกาศตัวแปรพอยน์เตอร์ ip
ip = msg1;              // กำหนดตำแหน่งหน่วยความจำอาเรย์ sale[0] ให้แก่ ip
```

หากสมมติกำหนดให้ msg1[] = " Test Pointer Array"; การเก็บข้อมูลของหน่วยความจำของตัวแปรอาเรย์ msg1 เป็นดังภาพที่ 7.5 และหากกำหนด ip = msg1; จะเป็นการกำหนดให้ตัวแปรพอยน์เตอร์ชี้ตำแหน่งหน่วยความจำอาเรย์ msg1[0]



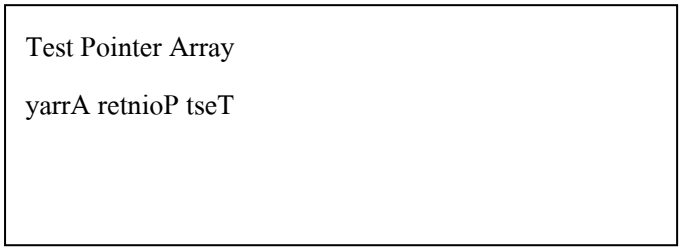
ภาพที่ 7.5 ลักษณะการจองและการกำหนดตำแหน่งหน่วยความจำอาเรย์

ตัวอย่างที่ 7.2 โปรแกรมการใช้ตัวแปรพอยน์เตอร์ชี้ตำแหน่งหน่วยความจำตัวแปรอาเรย์

```
#include <stdio.h>
#include <string.h>
void main()
{
  char msg1[] = " Test Pointer Array";
  int index, str_length;
  char *ip;                 // ประกาศตัวแปรพอยน์เตอร์ ip
  clrscr();
```

```
 str_length  = strlen(msg1)-1;
  ip =msg1;                  // กำหนดให้พอยน์เตอร์ ip ชี้ตำแหน่ง msg1[0]
  printf("\n");
  for(index=0;index<=str_length;index++)
  {
    printf("%c",*ip);
    ip++;                    //  เพิ่มค่าตำแหน่งหน่วยความจำ
  }
 printf("\n ");
 for (index = str_length;index >=0; index--)
    printf("%c",msg1[index]);
    getch();
}
```

ผลการทำงานของโปรแกรมจะเห็นว่าสามารถเข้าถึงข้อมูลของตำแหน่งหน่วยความจำอาเรย์ *ip ที่ตำแหน่งที่เรียงชิดติดกันของอาเรย์ได้อย่างสะดวกและง่าย สามารถนำมาประยุกต์ในการเปิดค่าในตารางได้อย่างสะดวกมากขึ้น และการทำงาน ดังแสดงในภาพที่ 7.6



ภาพที่ 7.6 ผลการทำงานของโปรแกรมในตัวอย่างที่ 7.2

7.3 พอยน์เตอร์และอาร์กิวเมนท์ของฟังก์ชัน

ฟังก์ชันบางฟังก์ชันในภาษาซีจะมีการส่งค่าอากิวเมนท์ ให้กับฟังก์ชันประเภทรับค่าข้อมูลเข้าไปประมวลผลภายในฟังก์ชัน และมีการคืนค่ากลับออกมา ซึ่งหากฟังก์ชันใดที่มีการคืนค่าข้อมูลออกมามากกว่าหนึ่งค่า การนำพอยน์เตอร์เข้ามาช่วยจะมีความสะดวกมากขึ้น เช่น หากต้องการเขียนฟังก์ชันเพื่อสลับค่าของตัวแปร 2 ตัว ผลลัพธ์ที่ต้องการได้จากฟังก์ชันนี้จะมีค่าของตัวแปร 2 ตัวที่ทำการสลับค่ากัน หากเขียนฟังก์ชันทั่วไปจะไม่สามารถแก้ปัญหานี้ได้ เนื่องจากฟังก์ชันแต่ละฟังก์ชันจะคืนค่าได้เพียงค่าเดียว ดังนั้นจึงต้องใช้พอยน์เตอร์เข้ามาช่วย โดยการส่งตำแหน่งของตัวแปรทั้ง 2 ให้กับฟังก์ชันที่จะสลับค่าของทั้ง 2 ตัวแปร ผ่านทางตัวแปรพอยน์เตอร์ที่เป็นพารามิเตอร์ของฟังก์ชัน

ตัวอย่างที่ 7.3 โปรแกรมสลับค่าตัวแปร 2 ตัวโดยผ่านฟังก์ชัน swap() โดยรับค่าข้อมูล 2 ค่า และเมื่อทำงานเสร็จให้มีการสลับค่าของตัวแปรทั้ง 2 ตัว โดยมีลักษณะของโปรแกรมดังนี้

```
#include <stdio.h>
void  swap(int *, int *);
void  main()
{
  int x =5, y=10;
  printf("Before swap : x = %d , y = %d\n",x,y);
  swap(&x,&y);
  printf("After swap : x = %d, y = %d\n",x,y
}
void swap (int *px, int *py)
{
  int temp;
  temp = *px;
  *px = *py;
  *py = temp;
}
```

อาร์กิวเมนท์ที่เป็นประเภทพอยน์เตอร์จะช่วยให้ฟังก์ชันสามารถเปลี่ยนค่าให้กับตัวแปรที่ส่งเข้ามาได้ เนื่องจากอาร์กิวเมนท์นั้นจะเก็บตำแหน่งของตัวแปรที่ส่งเข้ามา เมื่อมีการเปลี่ยนแปลงค่าของอาร์กิวเมนท์ผ่านตัวดำเนินการ * ค่าของตัวแปรที่ส่งเข้ามาจะถูกเปลี่ยนค่าพร้อมกันในทันที

ตัวอย่างที่ 7.4 การเขียนโปรแกรมเพื่อคำนวณพื้นที่ของสี่เหลี่ยม โดยรับข้อมูลความกว้างและความยาวของรูปสี่เหลี่ยม โดยภายในฟังก์ชันก็จะได้ผลลัพธ์ขนาดพื้นของสี่เหลี่ยมให้แก่ตัวแปรแบบพอยน์เตอร์ โดยมีการกำหนดให้ประกาศโปรโตไทป์ฟังก์ชันดังนี้

```
void calRecArea(float, float, float *);
```

โดยพารามิเตอร์ตัวแรกเป็นความกว้าง พารามิเตอร์ตัวที่ 2 คือความยาว และพารามิเตอร์ตัวที่ 3 คือ พื้นที่ของรูปสี่เหลี่ยม ซึ่งมีลักษณะของการเขียนโปรแกรมดังนี้

```
#include <stdio.h>
void calRecArea(float, float, float *);
void main()
{
  float width, length, area;
  printf("\n Please Enter Width:");
  scanf("%f",&width);
  printf("\n Please Enter length:");
  scanf("%f",length);
  calRecArea(width,length,&area);
  printf("Rectangle area is %.2f",area);
}
void calRecArea(float w, float l, float *area)
{
  *area = w*l;
}
```

แบบฝึกหัดท้ายหน่วยเรียนที่ 7

1) จงเขียนโปรแกรมเพื่อคำนวณเลขยกกำลัง โดยรับค่าเลขฐานและเลขยกกำลังจากผู้ใช้ โดยมีการกำหนดโปรโตไทป์ที่กำหนดดังนี้

การรับข้อมูล void readData(float *, int *);

การคำนวณ float power(float *, int);

โดยข้อมุลที่เข้าสู่ฟังก์ชันทั้งสองคือ เลขฐานและเลขยกกำลังตามลำดับ

2) จงเขียนโปรแกรมเพื่อคำนวณหาค่าจ้างของพนักงาน โดยกำหนดให้ฟังก์ชันรับค่าจำนวนชั่วโมงการทำงาน และประเภทของงานที่พนักงานทำ โดยกำหนดให้อัตราค่าจ้างแต่ละประเภทดังนี้

ตารางที่ 7.1 ประเภทของพนักงาน

ประเภทของงาน	อัตราค่าจ้าง/ชั่วโมง
1	30
2	40
3	45
4	50

โดยหากผู้ใช้ป้อนประเภทงานนอกเหนือจากประเภทที่กำหนดให้แสดงข้อความ " Wrong Type" และกำหนดให้เขียนฟังก์ชันเพื่อรับข้อมูลประเภทของงาน และอัตราค่าจ้างต่อชั่วโมงไว้ในฟังก์ชันเดียวกัน โดยฟังก์ชันมีลักษณะการประกาศโปรโตไทป์ที่กำหนดดังนี้

float cal(int);

โดยที่พารามิเตอร์ที่ส่งค่าออกคือค่าจ้างที่ได้รับ และพารามิเตอร์ที่เข้าในฟังก์ชันคือประเภทของงาน

ใบงานที่ 12

การใช้พอยน์เตอร์

จุดประสงค์

1) ทดลองเขียนโปรแกรมด้วยตัวแปรพอยน์เตอร์
2) สังเกตการณ์กำหนดค่าเริ่มต้นให้แก่ตัวแปรพอยน์เตอร์

การทดลองที่ 12.1 การกำหนดค่าเริ่มแรกให้ตัวแปรพอยน์เตอร์

ตัวแปรพอยน์เตอร์ก็มีความคล้ายกับตัวแปรอื่นๆในภาษาซี ที่สามารถกำหนดค่าเริ่มแรกในขณะที่ประกาศตัวแปร เช่น

```
int var1;
int *ptr_var = &var1;
```

ซึ่งเป็นการประกาศตัวแปรพอยน์เตอร์ ptr_var พร้อมทั้งกำหนดค่าเริ่มต้นคือตำแหน่งของตัวแปร var1 ไปพร้อมๆกัน โดยมีความหมายคล้ายกับการประกาศตัวแปรและใช้คำสั่งดังนี้

```
int var1;
int *ptr_var;
ptr_var = &var1;
```

ซึ่งในการทดลองนี้จะแสดงการประกาศตัวแปรพร้อมทั้งกำหนดค่าเริ่มต้นให้ตัวแปรพอยน์เตอร์ชนิดสตริง หรือข้อความ โดยมีขั้นตอนการทดลองดังนี้

1) เปิดโปรแกรมคอมไพเลอร์ Turbo C และ สร้างไฟล์ใหม่
2) เขียนโปรแกรมลงในอีดิตเตอร์ตามตัวอย่างโปรแกรมดังนี้

```
#include <stdio.h>
#include<string.h>
void main(void)
{
  char *string_ptr = "Hello C is a very flexibility");
```

```
  int index,str_length;
  clrscr();
  str_length = strlen(string_ptr)-1;
  for(index = str_length; index>=0;index--)
    printf("%c",string_ptr[index]);
  printf("\n");
  printf(string_ptr);
}
```

3) บันทึกไฟล์ในเป็นไฟล์ program12_1.c

4) เลือกคำสั่ง Compile และให้บันทึกผล

5) เลือกคำสั่ง RUN และให้บันทึกผลการทดลอง

……

……

……

……

……

……

……

……

การทดลองที่ 12.2 ตัวแปรพอยน์เตอร์สู่ตัวแปรอาร์เรย์

ตัวแปรอาเรย์มีความเกี่ยวข้องสัมพันธ์กับตัวแปรพอยน์เตอร์คือ ชื่อตัวแปรอาเรย์ (Array identifier) จะมีค่าเท่ากับตำแหน่งหน่วยความจำของสมาชิกตัวแรกของตัวแปรอาเรย์ เช่น

```
int num[20];
int *ptr;
ptr = Num;   // มีค่าเท่ากับ ptr = &Num[0];
```

ซึ่งเป็นการกำหนดตำแหน่งแรกของตารางให้กับตัวชี้พอยน์เตอร์ เมื่อต้องการเข้าถึงข้อมูลในลำดับถัดไปก็ให้เพิ่มค่าลำดับในตัวแปรพอยน์เตอร์หรือตัวชี้ เช่น

```
int num[20];
int *ptr,y;
ptr = Num;
y = *ptr;                  //  y = Num[0];
y = *(ptr+1);              //  y = Num[1];
y = *(ptr+2);              // y = Num[2];
```

ทั้งนี้การใช้ตัวแปรพอยน์เตอร์ชี้ตำแหน่งของอาเรย์ในตำแหน่งต่างๆ มีการทดลองดังนี้

1) เปิดโปรแกรมคอมไพเลอร์ Turbo C และ สร้างไฟล์ใหม่

2) เขียนโปรแกรมลงในอีดิตเตอร์ตามตัวอย่างโปรแกรมดังนี้

```
#include<stdio.h>
#include<conio.h>
int my_array[] = {1, 25, 15, 37, -5, 99};
int *ptr;
void main(void)
{
  int i;
  ptr = my_array;
  clrscr();
   for(i=0;i<6;i++)
     {
        printf("\n My array [%d] = %d",i,my_array[i]);
```

```
            printf(" = ptr+%d =%d",i,*(ptr+i));
        }
    }
```

3) บันทึกไฟล์ในเป็นไฟล์ program12_2.c

4) เลือกคำสั่ง Compile และให้บันทึกผล

5) เลือกคำสั่ง RUN และให้บันทึกผลการทดลอง

……

……

……

……

……

6) สรุปผลการทดลอง

.....

……

……

……

……

……

……

……

……

งานที่มอบหมาย

1) จงเขียนโปรแกรมแก้ปัญหาโจทย์ข้อ 2 ในแบบฝึกหัดท้ายบทเรียนที่ 7

ใบงานที่ 13

การเปิดข้อมูลจากตาราง

จุดประสงค์

1) ทดลองนำตัวแปรพอยน์เตอร์มากำหนดเป็นตาราง
2) สังเกตการณ์นำตัวแปรพอยน์เตอร์ค้นหาค่าข้อมูลในตาราง

ประยุกต์การเปิดตารางเทียบค่าแรงดันไฟฟ้ากับอุณหภูมิจากตารางเทอร์โมคัปเปิ้ล มาตรฐานชนิด K

การได้มาของค่าข้อมูลต่างๆ บางครั้งไม่สามารถนำสมการทางคณิตศาสตร์มาคำนวณหาค่าได้ เนื่องจากลักษณะของผลลัพธ์ไม่ได้เป็นเชิงเส้น (Non Linier) ซึ่งข้อมูลเหล่านั้นได้จากผลการทดลองในห้องปฏิบัติการแล้วบันทึกไว้เป็นตาราง เช่น ค่าแรงดันไฟฟ้าจากเทอร์โมคัปเปิลที่อุณหภูมิต่างๆ เทียบกับอุณหภูมิ 0 องศาเซลเซียส ซึ่งหากต้องการรู้ว่าที่จุดวัดอุณหภูมิ 30 ^{o}C เทียบกับอุณหภูมิ 0 ^{o}C เทอร์โมคัปเปิลมาตรฐาน ชนิด K จะผลิตแรงดันไฟฟ้าได้กี่มิลลิโวลท์ ซึ่งจำเป็นจะต้องทำการเปิดข้อมูลจากตารางของเทอร์โมคัปเปิล ชนิด K ดังตัวอย่างในตารางที่ 7.2 จึงจะสามารถบอกได้ว่าเทอร์โมคัปเปิลจะสร้างแรงดันไฟฟ้าเท่าใด ซึ่งหากนำข้อมูลเหล่านั้นมาไว้ในโปรแกรมเป็นลักษณะของตัวแปรอาเรย์ ซึ่งมีลักษณะคล้ายกับตาราง แล้วเขียนโปรแกรมไปดึงข้อมูล ณ อุณหภูมิที่ต้องการ ก็จะทำให้ทราบได้ว่าที่อุณหภูมิดังกล่าว เทอร์โมคัปเปิลจะสร้างแรงดันไฟฟ้าเท่าใด เพื่อนำมาเปรียบเทียบกับเทอร์โมคัปเปิลที่ใช้งาน หรือการประยุกต์ใช้งานในลักษณะต่อไป

ตารางที่ 7.2 แรงดันไฟฟ้าจากเทอร์โมคัปเปิ้ลมาตรฐานชนิด K เทียบอุณหภูมิ 0 °C

แรงดันไฟฟ้าจากเทอร์โมคัปเปิ้ล ชนิด K (mV)										
	0	1	2	3	4	5	6	7	8	9
0	0.000	0.039	0.079	0.119	0.158	0.277	0.317	0.277	0.317	0.357
10	0.397	0.437	0.477	0.517	0.557	0.597	0.637	0.677	0.718	0.758
20	0.798	0.838	0.879	0.919	0.960	1.000	1.041	1.081	1.122	1.163
30	1.203	1.244	1.285	1.326	1.366	1.407	1.448	1.489	1.530	1.571
40	1.612	1.653	1.694	1.735	1.776	1.817	1.858	1.899	1.941	1.982

ดังนั้นการนำพอยน์เตอร์มาเป็นตัวชี้ข้อมูลของตำแหน่งในตารางจึงมีความเหมาะสมและเข้าถึงข้อมูลได้ง่าย โดยใช้ลำดับของอุณหภูมิไปบวกกับค่าเริ่มแรกหรือตำแหน่งแรกของตารางก็จะได้ข้อมูลที่ถูกต้องมาใช้ในการคำนวณอื่น ต่อไป

การทดลองที่ 13.1 การเปิดข้อมูลจากตาราง

การนำข้อมูลจากตารางไปไว้ในโปรแกรมจะต้องนำไปไว้ในตัวแปรประเภทอาเรย์ ทั้งนี้จะใช้อาเรย์ชนิด 1 มิติเนื่องจากอุณหภูมิการการเรียงลำดับจาก 0 C ไปยัง 50 C อย่างต่อเนื่อง ซึ่งสามารถสร้างตารางได้ดังนี้

```
float table_K[50]={0.0, 0.039, 0.079,0.119, 0.158, 0.198, 0.238, 0.277, 0.317, 0.357,
                   0.397, 0.437, 0.477, 0.517, 0.557, 0.597, 0.637, 0.677, 0.718, 0.758,
                   0.798, 0.838, 0.879, 0.919, 0.960, 1.0, 1.041, 1.081, 1.122, 1.163,
                   1.203, 1.244, 1.285, 1.326, 1.366, 1.407, 1.448, 1.489, 1.530, 1.571,
                   1.612, 1.653, 1.694, 1.735, 1.776, 1.817, 1.858, 1.899, 1.941, 1.982,
                   1.612};
```

ซึ่งหากต้องการข้อมูลแรงดันไฟฟ้าที่อุณหภูมิใดก็ให้นำค่าลำดับอุณหภูมิไปบวกกับค่าเริ่มแรกของตารางที่พอยน์เตอร์ชี้อยู่ก็จะได้ข้อมูลแรงดันของอุณหภูมินั้นๆ เช่น

```
int *ptr_table;
float volt;
ptr_table = table_K;
volt  = *(ptr_table+temp);
```

ทั้งนี้สามารถดูลักษณะการเขียนโปรแกรมมีขั้นตอนการทดลองดังนี้

1) เปิดโปรแกรมคอมไพเลอร์ Turbo C และ สร้างไฟล์ใหม่

2) เขียนโปรแกรมลงในอีดิตเตอร์ตามตัวอย่างโปรแกรมดังนี้

```
#include <stdio.h>
#include<string.h>
float table_K[51]={0.0, 0.039, 0.079,0.119, 0.158, 0.198, 0.238, 0.277, 0.317, 0.357,
                   0.397, 0.437, 0.477, 0.517, 0.557, 0.597, 0.637, 0.677, 0.718, 0.758,
                   0.798, 0.838, 0.879, 0.919, 0.960, 1.0, 1.041, 1.081, 1.122, 1.163,
                   1.203, 1.244, 1.285, 1.326, 1.366, 1.407, 1.448, 1.489, 1.530, 1.571,
                   1.612, 1.653, 1.694, 1.735, 1.776, 1.817, 1.858, 1.899, 1.941, 1.982,
                   1.612};
float *ptr_table,volt;
int temp;
void main(void)
{
 char ch;
 do{
  ptr_table = table_K;
  clrscr();
  printf(" Please Enter the Temperature of Type K = ");
  scanf("%d",&temp);
  volt = *(ptr_table+temp);
  printf(" The voltage of TC type K = %.3f mV",volt);
  printf("\n Do you want to exit (Y/N)?");
  scanf("%c",&ch);
  getch();
  }while(ch != 'n');
}
```

3) บันทึกไฟล์ในเป็นไฟล์ program13_1.c

4) เลือกคำสั่ง Compile และให้บันทึกผล

5) เลือกคำสั่ง RUN และให้บันทึกผลการทดลอง

……

……

……

……

……

6) สรุปผลการทดลอง

.....

……

……

……

……

……

……

……

งานที่มอบหมาย

1) จงเขียนโปรแกรมหาค่าแรงดันไฟฟ้าจากการวัดเทอร์โมคัปเปิ้ลมาตรฐานชนิด J