บทที่ 7
เรื่อง Pointer

วัตถุประสงค์

1. เพื่อให้นักศึกษาเข้าใจหลักการทำงานของตัวแปร Pointer
2. เพื่อให้นักศึกษาสามารถสร้างและใช้ตัวแปร Pointer ได้อย่างมีประสิทธิภาพ

พอยน์เตอร์(Pointer)

เป็นตัวแปรที่ใช้ชี้บอกตำแหน่งหรือที่อยู่ของตัวแปรอื่นที่เก็บค่าข้อมูล เช่น ตัวแปร หรืออะเรย์ที่เก็บข้อมูลตัวเลขหรือตัวอักษร โดยตัวแปรพอยน์เตอร์ที่ใช้ต้องมีประเภทข้อมูลในการชี้เหมือนกัน และจากประโยชน์ที่มีมากมายของพอยน์เตอร์ จึงมีการใช้พอยน์เตอร์ในโปรแกรมภาษา C++ กันมาก ลักษณะของพอยน์เตอร์นั้นใกล้เคียงกับอะเรย์ และเป็นวิธีหนึ่งที่เราสามารถนำมาอ้างอิงสมาชิกอะเรย์ได้

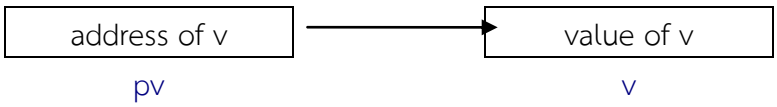
พอยน์เตอร์(Pointer) คือตัวแปรที่ชี้ไปยังที่ๆหนึ่งของหน่วยความจำคอมพิวเตอร์ ฉะนั้นตัวแปรพอยน์เตอร์คือ ตัวแปรที่เก็บค่าเอดแดรส(Address) ที่ชี้หรืออ้างอิงไปยังตัวแปรอื่นๆ หรือชี้ไปยังหน่วยความจำที่จอง ด้วยโอเปอร์เรเตอร์ new

แอดแดรส(Address) คือตำแหน่งการเก็บข้อมูลในหน่วยความจำ ซึ่งอาจจะเป็นของตัวแปร หรือที่จองด้วยโอเปอร์เรเตอร์ new

ตัวดำเนินการยูนารีที่ใช้งานกับพอยน์เตอร์ คือ

&	เป็นตัวดำเนินการที่ให้เป็นค่าของที่อยู่ (address operator)
*	เป็นตัวดำเนินการโดยอ้อมที่ให้ค่าของข้อมูล (indirection operator)

สมมติให้ v เป็นตัวแปรเก็บข้อมูล และให้ pv เป็นตัวแปรเก็บที่อยู่ของ v โดยจะเรียก pv ว่าเป็นตัวแปรพอยน์เตอร์ สามารถแสดงความสัมพันธ์ v และ pv ได้ดังรูป



ฉะนั้นจะได้ pv = &v และ v มีค่าเท่ากับ *pv

ตัวอย่าง

```
p = &temp;
```

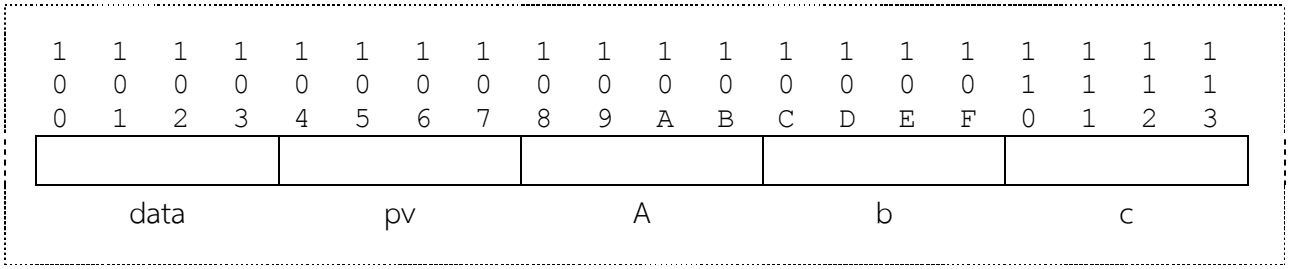
```
x = *p;
```

```
*p = 27 + y;
```

ความแตกต่างระหว่าง variable และ pointer

```
int     data= 30, *pv;
int     a, b, c;

pv = &data;          /* address of data is assigned to pv          */
a = pv;              /* contents (address of data) of pv is assigned */
                     /* to variable to a                           */
b = &pv              /* address of pv is assigned to variable b    */
c = *pv;             /* contents(value) at where the pv is indicating */
                     /* is assigned to variable c                  */
```



การประกาศพอยน์เตอร์ (Pointer Declaration)

ตัวแปรพอยน์เตอร์ก็เป็นเช่นเดียวกับตัวแปรอื่นๆที่ต้องมีการประกาศก่อนจึงจะสามารถใช้งานได้ การประกาศตัวแปรพอยน์เตอร์จะต้องนำหน้าชื่อตัวแปรด้วยเครื่องหมายดอกจัน (*) ซึ่งจะเป็นการบอกว่าเป็นตัวแปรแบบพอยน์เตอร์

รูปแบบ data-type *ptvar ;

ptvar คือชื่อของพอยน์เตอร์ และ data-type คือประเภทข้อมูลที่ต้องการชี้

ตัวอย่าง

```
int  v;
int  *pv;
pv = &v;
```

ตัวอย่างแสดงความสัมพันธ์ระหว่าง Pointer กับ Address ของตัวแปร

```
#include <iostream>
using namespace std;
int  main()
{
        int   i, *iptr = &i;
        cout << "\nEnter an integer value : ";
        cin >>  *iptr;
        cout << "\nThe value enter is " <<  i;
        return(0);
}
```

การทดลองที่ 7-1 แสดงการใช้ Pointer ชี้ไปยังตำแหน่งหน่วยความจำของตัวแปรและการอ้างใช้ค่าที่ตัวแปร pointer เก็บอยู่

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5          int  x=5, y, z=4;
6          int  *xptr, *yptr, *zptr;          /*       pointor to integer                */
7          xptr = &x;                         /*       assign address of x to xptr */
8          yptr = &y;                         /*       assign address of y to yptr */
9          zptr = &z;                         /*       assign address of z to zptr */
10         y = *xptr;                         /*       assign value of x to y  */
11         cout << "\nx = " << x <<  "  &x = " << &x;
12         cout << "  xptr = " << xptr << "  *xptr = " << *xptr;
13         cout  << "\ny = " << y << "  &y = " << &y ;
14         cout << "  yptr = " << yptr << "  *yptr = " << *yptr;
15         cout << "\nz = " << z << "  &z = " << &z ;
16         cout << "  zptr = " << zptr << "  *zptr = " <<  *zptr;
17         x = 2*(z+5);                       /*  ordinary expression                */
18         y = 2*(*zptr+5);                   /*       equivalent expression */
19         cout << "\nx = " << x << "  y =  " << y << "  z = " << z << endl;
20         return(0);
21 }
```

บันทึกผลการทดลอง

การทดลองที่ 7-2 แสดงขนาดของตัวแปร Pointer และขนาดของ *pointer

```
1  #include <iostream>
2  using  namespace std;
3  int main()
4  {
5          int    *iptr, i;
6          char   *cptr, c;
7          float  *fptr, f;
8          double *dptr, d;
9          iptr = &i;          cptr = &c;          fptr = &f;      dptr = &d;
```

```
10          cout << "Size of integer pointer is " << sizeof(iptr) << endl;
11          cout << "Size of char pointer is " << sizeof(cptr) << endl;
12          cout << "Size of float pointer is " << sizeof(fptr) << endl;
13          cout << "Size of double pointer is " << sizeof(dptr) << endl;
14          cout << "Size of *integer pointer is " << sizeof(*iptr) << endl;
15          cout << "Size of *char pointer is " << sizeof(*cptr) << endl;
16          cout << "Size of *float pointer is " << sizeof(*fptr) << endl;
17          cout << "Size of *double pointer is " << sizeof(*dptr) << endl;
18          return(0);
19  }
```

บันทึกผลการทดลอง

คำถาม ตัวแปร Pointer ที่มีชนิดข้อมูลแตกต่างกันจะมีขนาดของตัวแปรต่างกันหรือไม่ และมีขนาดเท่าไร

การส่งพอยน์เตอร์ให้กับฟังก์ชัน (Passing pointer to function)

ในการส่งผ่านตัวแปรพอยน์เตอร์เพื่อเป็นอาร์กิวเมนต์ให้กับฟังก์ชัน เราเรียกการใช้งานพอยน์เตอร์ในลักษณะนี้ว่า การส่งผ่านอาร์กิวเมนต์โดยการอ้างอิง (pass by reference)

การใช้ Increment operator (++) และ Decrement operator(--) ทำงานกับตัวแปรพอยน์เตอร์

การทดลองที่ 7-3 แสดงการใช้ตัวแปร Pointer กับฟังก์ชั่น

```
1   #include <iostream>
2   using namespace std;
3
4   void scan_line(char line[ ], int *pu, int *pl, int *pd, int *pw, int *po);
5   int main()
6   {
7           char  line[80];
8           int uppers, lowers, digits, whitespc, others;
9
```

```
10          uppers = lowers = digits = whitespc = others = 0;
11          cout << "\nEnter a line of text below :\n";
12          cin.getline(line,79);
13          scan_line(line,&uppers,&lowers,&digits,&whitespc,&others);
14          cout << "\nNumber of uppers character : " << uppers;
15          cout << "\nNumber of lower character : " << lowers;
16          cout << "\nNumber of digits : " << digits;
17          cout << "\nNumber of whitespace character : " << whitespc;
18          cout << "\nNumber of other character :  " << others;
19          cout << endl;
20          return(0);
21  }
22
23  void  scan_line(char line[], int *pu, int *pl, int *pd, int *pw, int *po)
24  {
25          int index = 0;
26          char  c;
27          while ((c = line[index++]) != '\0')
28          {
29                  if (isupper(c)) ++ *pu;
30                  else if (islower(c)) ++ *pl;
31                  else if (isdigit(c)) ++ *pd;
32          else if (isspace(c)) ++ *pw;
33          else ++ *po;
34          }
35  }
```

บันทึกผลการทดลอง

การทดลองที่ 7-4 แสดงการใช้ Pointer กับการคืนค่าของฟังก์ชั่น

```
1  #include <iostream>
2  using namespace std;
3  char  *month(int n);
4  int main()
5  {
6          int  n;
7          bool flag = true;
8          char  text[10], *str;
9          str = (char *) malloc(12);        // alloc memory
10         do {
11                 cout << "\nInput month number(0-quit) :";
12                 cin >> text;
13                 n = atoi(text);
14                 if (n != 0)  {
15                         if (n > 0 && n <13) {
16                                 strcpy(str,month(n));
17                                 cout << "Month name : " << str << endl;
18                         }
19                         else printf("Month number error\n");
20                 }
21                 else flag = false;
22         } while (flag);
23         return(0);
24 }
25
26 char  *month(int n)
27 {
28         char  m[12][10] = {"January","February","March","April","May",
29                 "June","July","August","September","October",
30                 "November","December"};
31         return(m[n-1]);
32 }
```

บันทึกผลการทดลอง

พอยน์เตอร์กับอาเรย์ 1 มิติ (Pointer and 1 dimensional array)

โดยปกติชื่อของอาเรย์ก็คือพอยน์เตอร์ที่ชี้ไปยังสมาชิกตัวแรกของอาเรย์ ซึ่งมีวิธีเขียนที่อยู่ของสมาชิกของอาเรย์ได้ 2 วิธี คือ

1) เขียนด้วยเครื่องหมาย & แล้วตามด้วยการเขียนสมาชิกอาเรย์ตามปกติ

&x[0] , &x[1] , &x[2], ... หรือ &x[i]

2) เขียนเป็นนิพจน์โดยนำเอาซับสคริปต์มาบวกกับชื่ออาเรย์

x , x+1 , x+2 , ... หรือ x + i

เนื่องจาก &x[i] และ (x+i) ต่างก็แทนที่อยู่ของสมาชิกตัวที่ i ของอาเรย์ x จึงเรียก i เมื่อนำมาใช้ว่าออฟเซต (offset) ดังนั้น x[i] และ *(x+i) แทนค่าข้อมูลที่เก็บอยู่ในที่อยู่นั้น ฉะนั้นผู้เขียนโปรแกรมสามารถเขียนได้ทั้งสองแบบ

การทดลองที่ 7-5 แสดงการใช้ Pointer กับอาเรย์ 1 มิติ

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5          float A[10]={10,11,12,13,14,15,16,17,18,19};
6          float *ptr;
7          ptr = A;
8          cout << "\nStart address of array A = " << A;
9          cout << "\nEnd address of array A = " << A+9 << endl;
10         for (int i = 0 ; i < 10 ; i++)
11         {
12                 cout << "\n i = " << i << "  A[i] = " << A[i];
13                 cout << "  *(ptr+i) = "  << *(ptr+i);
14                 cout << "  &A[i] = " << &A[i] << "  ptr+i = " << ptr+i << endl;
15         }
16         return(0);
17 }
```

บันทึกผลการทดลอง

การขอหรือจัดสรรหน่วยความจำ (Dynamic memory allocate)

สำหรับตัวแปรพอยน์เตอร์เมื่อต้องการใช้งานจะต้องมีการร้องขอพื้นที่หน่วยความจำก่อนจึงจะทำงานได้ แต่ระบบจะไม่ขอหรือจัดสรรให้ได้อัตโนมัติต้องเขียนคำสั่งเพื่อขอเอง โดยการใช้คำสั่ง **new** ไม่เหมือนกับการประกาศตัวแปร array ที่ระบบจะจัดการขอให้โดยอัตโนมัติ และสามารถคืนหน่วยความจำด้วยคำสั่ง **delete**

ตัวอย่าง

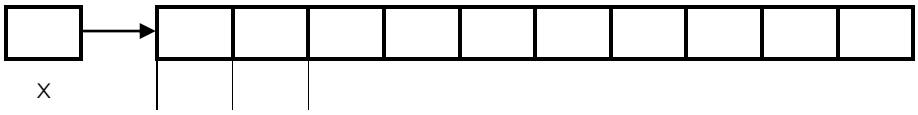
ต้องการใช้งานตัวแปร x เป็น 1 มิติขนาด 10 สมาชิกโดยเก็บข้อมูลแบบ integer

เมื่อใช้ array เขียนดังนี้ int x[10];

เมื่อใช้ pointer เขียนดังนี้ int *x;

แต่หากใช้เป็น pointer จะยังไม่สามารถใช้งานได้ต้องทำการขอหน่วยความจำก่อน ด้วยคำสั่งnew

```
x = new int[10];
```



การทดลองที่ 7-6 การใช้งานคำสั่ง new และ delete กับพอยน์เตอร์ในลักษณะอะเรย์ 1 มิติ

```
1  #include <iostream>
2  #include <iomanip>
3  #include <time.h>
4  using namespace std;
5  void  Gen_Number(int *x,int n);
6  void  Reorder(int *x, int n);
7  int main()
8  {
9          int i, n , *x;
10
11         cout << "\nHow many numbers will be entered ? ";
12         cin >> n;
13         /* Allocate Memory */
14         x = new int [n];
15         Gen_Number(x, n);     /* generate data number by random  */
16         cout << "\n\nBefore sort of array:\n";
17         for ( i = 0 ; i < n ; i++) {
18                 cout << "\ni = " << setw(2) << i << "   *(x+i) = ";
19                 cout << setw(2) << *(x+i) << "   x[i] = " << setw(2) << x[i];
20         }
21         /* reorder all array elements  */
22         Reorder( x, n);
23         cout << "\n\nReordered list of numbers:\n";
24         for ( i = 0 ; i < n ; i++){
25                 cout << "\ni = " << setw(2) << i << "   *(x+i) = ";
26                 cout << setw(2) << *(x+i) << "   x[i] = " << setw(2) << x[i];
27         }
28         cout << endl;
```

```
29          delete [] x;
30          return(0);
31  }
32
33  void  Gen_Number(int *x,int n)
34  {
35          int  i;
36          srand( time(0));
37          for ( i = 0 ; i < n ; i++) x[i] = rand() % 100;
38  }
39
40  void  Reorder(int *x, int n)
41  {
42          int  i,item, temp;
43          for( item = 0; item < n-1 ; ++item) {
44                  for( i = item + 1; i < n ; i++)
45                          if (*(x+i) < *(x+item))   {
46                                  temp = *(x+item);
47                                  *(x+item) = *(x+i);
48                                  *(x+i) = temp;
49                          }
50          }
51  }
```

บันทึกผลการทดลอง

แบบฝึกหัดท้ายบท

1. ให้นักศึกษาเขียนโปรแกรมเพื่อแปลงข้อมูลวันที่จากตัวเลขให้เป็นข้อความ โดยเขียนเป็นฟังก์ชั่นในการทำงานแบบส่งค่ากลับคืนที่ชื่อฟังก์ชั่น

   ตัวอย่าง Input Date(dd/mm/yyyy) : 15/09/2000
   Format 1: September 15, 2000
   Format 2: 15 Sep 2000

2. ให้นักศึกษาเขียนโปรแกรมเพื่อคำนวณเกรดของนักศึกษาที่ได้และหาจำนวนนักศึกษาในแต่ละเกรด โดยโปรแกรมจะรับจำนวนนักศึกษาที่ต้องการทำงาน เพื่อนำมาสร้างตัวแปรอาเรย์ แล้วนำมามารับข้อมูลคะแนนของนักศึกษาหรือใช้การสุ่มค่าจาก 0 – 100 โดยให้สร้างฟังก์ชั่นเพื่อตรวจสอบเกรดที่ได้ตามคะแนนที่กำหนด และหาจำนวนความถี่ของแต่ละเกรดว่ามีจำนวนกี่คน โดยใช้หลักการ Frequency Distributions พร้อมแสดงรายงานออกที่หน้าจอ

   กำหนดให้

คะแนน	เกรด
0 - 49	F
50 - 59	D
60 - 69	C
70 - 79	B
80 - 100	A

   หมายเหตุ ให้ใช้ตัวแปรพอยน์เตอร์ในการจัดเก็บข้อมูล