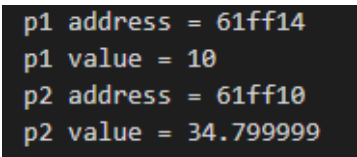
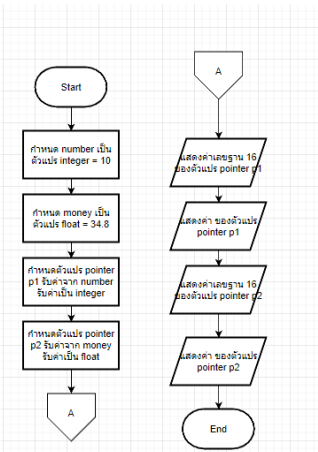


ตอนที่ 1 จงอธิบายความหมายพร้อมยกตัวอย่างประกอบ

จงเขียนคำอธิบาย ยกตัวอย่างประกอบ และวาดรูปประกอบตามความเข้าใจของคุณ

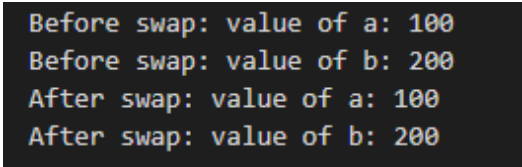
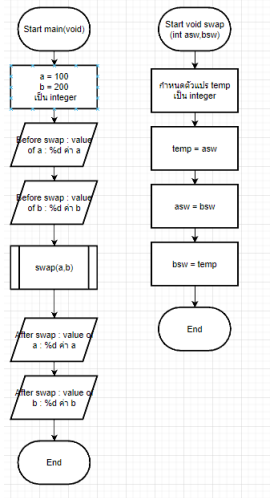
| ข้อที่ 1 จงอธิบายความหมายของ Pointer อย่างละเอียด และยกตัวอย่างการใช้งาน พร้อม Code ตัวอย่างการใช้งาน Pointer | |
|--|--|
| คำอธิบาย | Code ตัวอย่าง |
| <p>Pointer พูดให้เข้าใจอย่างง่ายคือเป็นตัวที่คอยชี้ทางให้กับที่อยู่ของข้อมูล เป็นตัวแปรที่เก็บค่าของที่อยู่ของข้อมูลของตัวแปร</p> <p>ทำให้เราสามารถเข้าถึงข้อมูลได้โดยตรงโดยใช้ที่อยู่ของหน่วยความจำ</p> <p>โดยมีเครื่องหมายที่สำคัญในการใช้ Pointer ได้แก่</p> <ul style="list-style-type: none"> Address ที่อยู่ข้อมูล (&) Dereference Operator ตัวดำเนินการประกาศ (*) Declaration Operator ตัวดำเนินการอ้างอิง (*) | <pre> 1 #include <stdio.h> 2 3 int main() 4 { 5 int number = 10; 6 float money = 34.8; 7 8 int *p1 = &number; 9 float *p2; 10 p2 = &money; 11 12 printf("p1 address = %x\n", p1); //Output ของ p1 ในรูปแบบเลขฐาน 16 คือการใช้ %x 13 printf("p1 value = %d\n", *p1); 14 printf("p2 address = %x\n", p2); //Output ของ p2 ในรูปแบบเลขฐาน 16 คือการใช้ %x 15 printf("p2 value = %f\n", *p2); 16 return 0; 17 } </pre> |
| ผลลัพธ์ของ Code (Captureพร้อมแปะรูป) | Flow chart ของ Code ตัวอย่าง |
|  <pre> p1 address = 61ff14 p1 value = 10 p2 address = 61ff10 p2 value = 34.799999 </pre> |  <pre> graph TD Start([Start]) --> A1[/A/] A1 --> N1[กำหนด number เป็น ตัวแปร integer = 10] N1 --> N2[กำหนด money เป็น ตัวแปร float = 34.8] N2 --> N3[กำหนดตัวแปร pointer p1 รับค่าเป็น integer] N3 --> N4[กำหนดตัวแปร pointer p2 รับค่าเป็น float] N4 --> A2[/A/] A2 --> P1[/แสดงค่า ของตัวแปร pointer p1/] P1 --> P2[/แสดงค่า ของตัวแปร pointer p2/] P2 --> End([End]) </pre> |

| ข้อที่ 2 จงสร้าง Pointer จำนวน 1 ตัวที่ชี้ Array ไม่จำกัดแถว แถวละ 4 Column และยกตัวอย่างการใช้งาน พร้อม Code ตัวอย่างการใช้งาน | |
|---|---|
| คำอธิบาย | Code ตัวอย่าง |
| <p>การใช้ Pointer มีประโยชน์กับ Array เพราะ Array มีการเก็บข้อมูลเป็นชุดอันดับ และเรียงต่อกัน</p> <hr style="border-top: 1px dashed black;"/> <p>จากโค้ดเป็นการใช้ Pointer กับ ตัวแปรอาร์เรย์ ซึ่งด้วยคำสั่ง Pointer ทำให้เราสามารถเลื่อนตำแหน่งของค่าที่ถูกจัดเรียงตามลำดับใน Array ได้</p> <p>โดยเรากำหนดตัวแปร Pointer ชื่อ myPointer ให้มีค่าเป็น number โดยสามารถเลื่อนตำแหน่งได้โดยการ + หรือ - ค่า Pointer และแสดงผลลัพธ์ในตำแหน่ง Array ที่เราต้องการ</p> | <pre style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> 1 #include <stdio.h> 2 3 int main() 4 { 5 int number[5] = {10, 20, 30, 40, 50}; 6 int *myPointer = &number; 7 8 printf("1st Place -> %d\n", *myPointer); 9 myPointer++; 10 printf("2nd Place -> %d\n", *myPointer); 11 myPointer += 3; 12 printf("5nd Place -> %d\n", *myPointer); 13 myPointer--; 14 printf("4th Place -> %d\n", *myPointer); 15 return 0; 16 }</pre> |
| <p>ผลลัพธ์ของ Code (Captureพร้อมแปะรูป)</p> <div style="background-color: #333; color: white; padding: 10px; text-align: center; margin: 10px 0;"> <p>1st Place -> 10</p> <p>2nd Place -> 20</p> <p>5nd Place -> 50</p> <p>4th Place -> 40</p> </div> | <p>Flow chart ของ Code ตัวอย่าง</p> <pre> graph TD Start([Start]) --> Init[กำหนด เป็นตัวแปร Array ขนาด 5 รับค่า int ได้แก่ 10 20 30 40 50] Init --> Decl[กำหนดตัวแปร Pointer ชื่อ myPointer ให้มีค่า number] Decl --> A{A} A -- Yes --> Print1[/1st Place -> %d\n โดยแสดงค่าจาก Pointer *myPointer/] Print1 --> Inc1[myPointer++ เลื่อนตำแหน่ง +1] Inc1 --> A A -- No --> Print2[/2nd Place -> %d\n โดยแสดงค่าจาก Pointer *myPointer/] Print2 --> Inc3[myPointer เลื่อนตำแหน่ง +3] Inc3 --> Print3[/5th Place -> %d\n โดยแสดงค่าจาก Pointer *myPointer/] Print3 --> Dec1[myPointer เลื่อนตำแหน่ง -1] Dec1 --> Print4[/4th Place -> %d\n โดยแสดงค่าจาก Pointer *myPointer/] Print4 --> End([End]) </pre> |

| ข้อที่ 3 จงอธิบายเรื่อง Pointer Functionยกตัวอย่างการใช้งาน พร้อม Code ตัวอย่างการใช้งาน | |
|--|--|
| คำอธิบาย | Code ตัวอย่าง |
| <p>เป็นฟังก์ชันที่มีโครงสร้างเหมือนฟังก์ชันอื่นทั่วไป</p> <p>แต่ประกาศชื่อฟังก์ชันเป็น Pointer โดยต้องมีวงเล็บครอบไว้ เป็น</p> <p>typename(*functionname)(parameter);</p> <p>จากโค้ด</p> | <pre>#include <stdio.h> void fun(int a) { printf("ค่าที่ Pointer ชี้เป็น %d\n", a); } int main() { void (*fun_ptr)(int) = fun; // & removed fun_ptr(10); // * removed return 0; }</pre> |
| ผลลัพธ์ของ Code (Captureพร้อมแปะรูป) | Flow chart ของ Code ตัวอย่าง |
|  |  <pre>graph TD Start([Start]) --> J(()) J --> D{กำหนด Pointer Function ชื่อ fun_ptr รับค่า integer} D --> A[ค่า fun_ptr เป็น 10] A --> B[void fun รับค่า int a] B --> J StartVoid([Start Void fun]) --> I[/ค่าที่ Pointer ชี้เป็น a/] I --> End([End])</pre> |

| ข้อที่ 4 จงอธิบายเรื่อง Dynamic Array ยกตัวอย่างการใช้งาน พร้อม Code ตัวอย่างการใช้งาน | |
|---|--|
| คำอธิบาย | Code ตัวอย่าง |
| <p>Dynamic Array คืออาร์เรย์ที่สามารถเปลี่ยนขนาดได้</p> <p>ในขณะที่โค้ดกำลังทำงาน และสามารถเพิ่มองค์ประกอบได้อย่างต่อเนื่องที่ตำแหน่งสิ้นสุดของ Array Dynamic</p> | <pre> 1 #include <stdio.h> 2 3 int main(){ 4 5 int row,col; 6 printf("Input your row and column: "); 7 scanf("%d %d",&row,&col); 8 9 int *a; 10 a = new int(row*col) ; 11 12 for(int i = 0; i < row; i++){ 13 for(int j = 0; j < col; j++){ 14 printf("a[%d][%d] = ", i,j); 15 scanf("%d", &a[i * col + j]); 16 } 17 } 18 for(int i = 0; i < row * col ; i++){ 19 printf("%d\t", a[i]); 20 if((i+1)%col == 0){ 21 printf("\n"); 22 } 23 } 24 return 0 ; 25 }</pre> |
| ผลลัพธ์ของ Code (Captureพร้อมแปะรูป) | Flow chart ของ Code ตัวอย่าง |
| <p>Output</p> <pre> /tmp/qoq0ZMVlT8.o Input your row and column: 2 2 a[0][0] = 1 a[0][1] = 2 a[1][0] = 3 a[1][1] = 4 1 2 3 4</pre> | <pre> graph TD Start([Start]) --> Init[กำหนดตัวแปร p num เป็น integer] Init --> Input[/Enter the numbers ป้อนค่าไว้ใน num/] Input --> Create[สร้างระบบPointer ชื่อ Array รับค่า num] Create --> Input2[/Enter จำนวน numbers/] Input2 --> Loop1{p < num?} Loop1 -- Yes --> Read[รับค่าป้อนไว้ใน Array ตัวที่ Array[p]] Read --> Print[/Your numbers are/] Print --> Loop2{p < num?} Loop2 -- Yes --> PrintArray[/Array [p]/] PrintArray --> Inc[p++] Inc --> Loop1 Loop1 -- No --> End([End]) </pre> |

| ข้อที่ 5 จงอธิบายการส่งผ่านตัวแปรแบบ Pass by reference ยกตัวอย่างการใช้งาน พร้อม Code ตัวอย่างการใช้งาน | |
|---|--|
| คำอธิบาย | Code ตัวอย่าง |
| <p>Pass by Ref คือการส่งค่าไปยังฟังก์ชันที่ถูกเรียกใช้ โดยส่งเป็นค่าตำแหน่งที่อยู่ (Address) ของตัวแปรไปซึ่งหากภายในฟังก์ชัน มีการเปลี่ยนแปลงค่าของ Argument ที่ส่งไป ก็จะมีผลทำให้ค่านั้นเปลี่ยนไปด้วย</p> | <pre>#include <stdio.h> int main () { int a = 100; int b = 200; printf("Before swap, value of a : %d\n", a); printf("Before swap, value of b : %d\n", b); swap(&a, &b); printf("After swap, value of a : %d\n", a); printf("After swap, value of b : %d\n", b); return 0; } void swap(int *x, int *y) { int temp; temp = *x; //เก็บค่า pointer x ไว้ที่ temp *x = *y; //รับค่า pointer y เก็บไว้ที่ pointer x *y = temp; //รับค่า temp เก็บไว้ที่ pointer y เป็นการสลับค่ากัน return; }</pre> |
| ผลลัพธ์ของ Code (Captureพร้อมแปะรูป) | Flow chart ของ Code ตัวอย่าง |
| <div>Before swap, value of a : 100 Before swap, value of b : 200 After swap, value of a : 200 After swap, value of b : 100</div> | <pre> graph TD subgraph Main Start([Start]) --> Init[กำหนดตัวแปร a = 100 b = 200] Init --> PrintBefore[Before swap, value of a : %d เป็นค่า a] PrintBefore --> PrintBeforeB[Before swap, value of b : %d เป็นค่า b] PrintBeforeB --> CallSwap[swap(&a, &b)] CallSwap --> PrintAfterA[After swap, value of a : %d เป็นค่า a] PrintAfterA --> PrintAfterB[After swap, value of b : %d เป็นค่า b] PrintAfterB --> EndMain([End]) end subgraph SwapFunction StartSwap([Start void swap กำหนดตัวแปร Pointer x และ y]) --> DeclTemp[กำหนดตัวแปร temp เป็น integer] DeclTemp --> AssignTemp[กำหนดค่า temp = pointer x] AssignTemp --> AssignX[กำหนดค่า pointer x = pointer y] AssignX --> AssignY[กำหนดค่า pointer y = temp] AssignY --> EndSwap([End]) end </pre> |

| ข้อที่ 6 จงอธิบายการส่งผ่านตัวแปรแบบ Pass by value ยกตัวอย่างการใช้งาน พร้อม Code ตัวอย่างการใช้งาน | |
|--|--|
| คำอธิบาย | Code ตัวอย่าง |
| <p>Pass by value คือ การส่งค่าเป็น argument ของ function โดยค่าในฟังก์ชันจะไม่ส่งผลต่อตัวแปรนอกฟังก์ชัน</p> | <pre> 1 void swap(int asw, int bsw) 2 { 3 int temp; 4 5 temp = asw; 6 asw = bsw; 7 bsw = temp; 8 9 return; 10 } 11 12 int main(void) 13 { 14 int a = 100; 15 int b = 200; 16 17 printf("Before swap: value of a: %d\n", a); 18 printf("Before swap: value of b: %d\n", b); 19 20 swap(a, b); 21 22 printf("After swap: value of a: %d\n", a); 23 printf("After swap: value of b: %d\n", b); 24 25 return 0; 26 } </pre> |
| ผลลัพธ์ของ Code (Captureพร้อมแปะรูป) | Flow chart ของ Code ตัวอย่าง |
|  <pre> Before swap: value of a: 100 Before swap: value of b: 200 After swap: value of a: 100 After swap: value of b: 200 </pre> |  <pre> graph TD subgraph Main_Function [int main(void)] StartMain([Start main(void)]) --> InitMain[a = 100 b = 200 เป็น integer] InitMain --> PrintBeforeMain[/Before swap: value of a: %d คำ a/] PrintBeforeMain --> PrintBeforeMain2[/Before swap: value of b: %d คำ b/] PrintBeforeMain2 --> CallSwap[swap(a, b)] CallSwap --> PrintAfterMain1[/After swap: value of a: %d คำ a/] PrintAfterMain1 --> PrintAfterMain2[/After swap: value of b: %d คำ b/] PrintAfterMain2 --> EndMain([End]) end subgraph Swap_Function [void swap(int asw, int bsw)] StartSwap([Start void swap int asw, bsw]) --> TempInit[กำหนดตัวแปร temp เป็น integer] TempInit --> TempAssign[temp = asw] TempAssign --> AswAssign[asw = bsw] AswAssign --> BswAssign[bsw = temp] BswAssign --> EndSwap([End]) end </pre> |