ใบงานที่ 6

เรื่อง Calculate Postfix

เสนอ

อาจารย์ ปิยพล ยืนยงสถาวร

จัดทำโดย

นายกฤษฎา วิริยา

65543206041-7

ใบงานนี้เป็นส่วนหนึ่งของรายวิชา โครงสร้างข้อมูลและขั้นตอนวิธี

หลักสูตรวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา

ประจำภาคที่ 1 ปีการศึกษา 2566

คำสั่ง/คำชี้แจง

- แสดงโค้ดโปรแกรมเป็นส่วนๆพร้อมทั้งอธิบาย
- แสดงผลการรันโปรแกรม พร้อมอธิบายการทำงาน
- สรุปผลการทดลอง

ลำดับดับขั้นการทดลอง

```
#include <stdio.h>                //use printf()
#include <conio.h>                //use getch()
#include <string.h>               //use string function
#include <math.h>                 //use power
#define MaxStack 40               // Set Max Operator Stack
char postfix1[80] = {"AB+C-d/"}; // Assign INFIX
float ValPostfix[80];             // Keep value of Postfix here
float ValOperandST[MaxStack];     // Operator stack size
int SP = 0;                       // Initial SP=0
```

- **#include <stdio.h>:** เป็นส่วนของโค้ดที่ใช้ระบุว่าโปรแกรมต้องการใช้ฟังก์ชันที่เกี่ยวข้องกับการป้อนออกข้อมูล
- **#include <conio.h>:** เป็นส่วนของโค้ดที่ใช้ระบุว่าโปรแกรมต้องการใช้ฟังก์ชันที่เกี่ยวข้องกับการควบคุมอินพุตทางคีย์บอร์ด getch()
- **#include <string.h>:** ใช้เพื่อนำเข้าไฟล์ส่วนของฟังก์ชันที่เกี่ยวข้องกับการจัดการกับสตริง (string) เช่น strlen() เพื่อหาความยาวของสตริง และ strcpy() เพื่อคัดลอกสตริง.
- **#include <math.h>:** ใช้เพื่อนำเข้าไฟล์ส่วนของฟังก์ชันทางคณิตศาสตร์ เช่น pow() เพื่อคำนวณค่ายกกำลัง.
- **#define MaxStack 40:** กำหนดค่าคงที่ MaxStack เป็น 40 ซึ่งเป็นขนาดสูงสุดของสแต็กแบบตัวดำเนินการ (Operator Stack) ที่ใช้ในโปรแกรม.
- **char postfix1[80] = {"AB+C-d/"}:** ประกาศตัวแปรชนิด char ชื่อ postfix1 และกำหนดค่าให้เป็นสตริง "AB+C-d/" ซึ่งเป็นนิพจน์ทางคณิตศาสตร์ในรูปแบบ Postfix (หลังนิพจน์) ที่ต้องการทำงานกับ.
- **float ValPostfix[80]:** ประกาศตัวแปรชนิด float ชื่อ ValPostfix ที่ใช้เก็บค่าของนิพจน์ที่ได้จากการประมวลผล Postfix.
- **float ValOperandST[MaxStack]:** ประกาศตัวแปรชนิด float ชื่อ ValOperandST ที่ใช้เก็บค่าตัวแปรในสแต็กแบบตัวดำเนินการ (Operator Stack) โดยมีขนาดสูงสุดตามค่า MaxStack ที่กำหนดไว้.

- **int SP = 0:** ประกาศตัวแปรชนิด int ชื่อ SP (Stack Pointer) และกำหนดค่าเริ่มต้นให้เป็น 0 ซึ่งเป็นตัวชี้ที่ใช้บ่งชี้ตำแหน่งล่าสุดในสแต็กแบบตัวดำเนินการ.

```
16 void push(float ValOperand) { // PUSH Function
17   if (SP == MaxStack) // Check Stack FULL?
18     printf("ERROR STACK OVER FLOW!!!...\n");
19   else {
20     SP = SP + 1;                    // Increase SP
21     ValOperandST[SP] = ValOperand; // Put data into Stack
22   }
23 }
```

push(float ValOperand): นี่คือฟังก์ชันที่ใช้ในการเพิ่มค่า ValOperand เข้าสู่สแต็ก (stack) ของตัวแปร ValOperandST.

- เมื่อเรียกใช้ push() ฟังก์ชันจะตรวจสอบว่าสแต็กเต็มหรือไม่ โดยตรวจสอบค่าของตัวแปร SP ว่าเท่ากับ MaxStack หากเต็ม จะแสดงข้อความ "ERROR STACK OVER FLOW!!!..." เพื่อแจ้งเตือนผู้ใช้.
- หากสแต็กยังไม่เต็ม ฟังก์ชันจะทำการเพิ่มค่าของ SP ขึ้น 1 หน่วย (SP = SP + 1) เพื่อเลื่อนตำแหน่งสำหรับการเก็บข้อมูลใหม่ในสแต็ก.
- จากนั้น ค่าของ ValOperand จะถูกเก็บไว้ในอาร์เรย์ ValOperandST ในตำแหน่งที่ SP กำหนด.

```
25 float pop() {// POP Function
26   float ValOperand;
27   if (SP != 0) {// Check Stack NOT EMPTY?
28     ValOperand = ValOperandST[SP]; // Get data from Stack
29     SP--;                          // Decrease SP
30     return (ValOperand);           // Return data
31   }
32   else
33     printf("\nERROR STACK UNDER FLOW!!!...\n");
34 }
```

pop(): นี่คือฟังก์ชันที่ใช้ในการเอาค่าจากสแต็ก (stack) ของตัวแปร ValOperandST และลดค่าของ SP ลงหนึ่งหน่วย.

- เมื่อเรียกใช้ pop() ฟังก์ชันจะตรวจสอบว่าสแต็กว่างเปล่าหรือไม่ โดยตรวจสอบค่าของตัวแปร SP ว่าเท่ากับ 0 หากว่างเปล่า จะแสดงข้อความ "ERROR STACK UNDER FLOW!!!..." เพื่อแจ้งเตือนผู้ใช้.

- หากสแต็กไม่ว่างเปล่า ฟังก์ชันจะดึงค่าจากตำแหน่ง SP ของอาร์เรย์ ValOperandST และนำค่านั้นไปใช้ในการคำนวณ.
- ต่อมา ค่าของ SP จะถูกลดลงหนึ่งหน่วย (SP--) เพื่อเลื่อนตำแหน่งสำหรับการเข้าถึงค่าในสแต็ก.

```
void CalPostfix(char postfix[80]) {
  float pop1, pop2, value;
  int i, len;
  char ch;
  len = strlen(postfix);
  printf("Postfix = %s\n", postfix);
  for (i = 0; i <= len - 1; i++) { // Assign data to OPERAND
    ch = postfix[i];               // Split Character for assign data
    if (strchr("+-* /^", ch) == 0) { // Check Is OPERAND?
      printf("\nAssign Number to %c : ", ch);
      scanf("%f", &ValPostfix[i]); // Read data from KBD and direct to Value of OPERAND in Array
    }
  }
  for (i = 0; i <= len - 1; i++) { // Calculate Value of POSTFIX
    ch = postfix[i];               // Split Character for prepare to STACK64
    if (strchr("+-* /^", ch) == 0) // Check Is OPERAND?
      push(ValPostfix[i]); // push value of OPERAND to Stack
    else {
      pop1 = pop(); // Pop 1st
      pop2 = pop(); // pop 2nd
      switch (ch)
      {
      case '+':
        value = pop2 + pop1; // Calculate
        push(value);         // Push value to Stack
        break;
      case '-':
        value = pop2 - pop1;
        push(value);
        break;
      case '*':
        value = pop2 * pop1;
        push(value);
        break;
      case '/':
        value = pop2 / pop1;
        push(value);
        break;
      case '^':
        value = pow(pop2, pop1);
        push(value);
        break;
      }
    }                          // End IF
  }                            // End IF
  printf("\nANS = %f", pop()); // Last value is ANSWER
} // End Fn.
```

CalPostfix(char postfix[80]): นี่คือฟังก์ชันหลักที่ใช้ในการคำนวณค่าของนิพจน์ Postfix โดยรับพารามิเตอร์ postfix ที่เป็นสตริง (string) ของนิพจน์ Postfix.

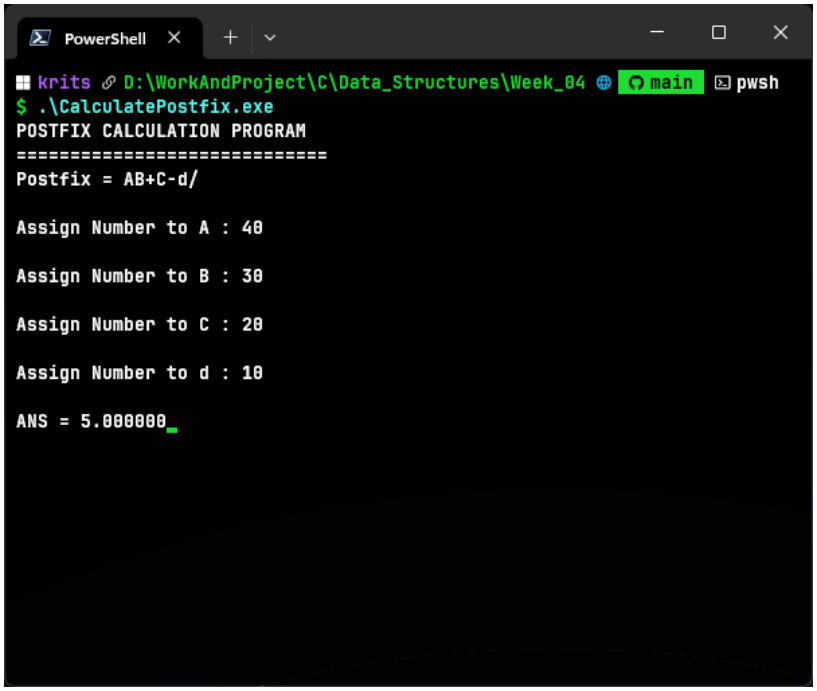
- ฟังก์ชันจะคำนวณความยาวของสตริง postfix ด้วยฟังก์ชัน strlen() เพื่อใช้ในการวนลูปสำหรับการดำเนินการในนิพจน์ Postfix.
- ฟังก์ชันจะแสดงข้อความ "Postfix = " ตามด้วยค่าของ postfix เพื่อแสดงนิพจน์ Postfix ที่ใช้ในการคำนวณ.
- ฟังก์ชันจะวนลูปเพื่อกำหนดค่าตัวเลขของ OPERAND จากผู้ใช้. และเก็บค่าตัวเลขนั้นลงในอาร์เรย์ ValPostfix.

- ในการคำนวณค่าของนิพจน์ Postfix เมื่อเจอ OPERAND จะใช้ฟังก์ชัน push() เพื่อเก็บค่า OPERAND ลงในสแต็ก.
- หากเจอ OPERATOR ฟังก์ชันจะใช้ฟังก์ชัน pop() เพื่อดึงค่า OPERAND สองตัวมาใช้ในการคำนวณตามตัวดำเนินการ.
- ฟังก์ชันจะใช้ switch เพื่อดำเนินการคำนวณตามตัวดำเนินการที่พบ เช่น +, -, *, /, ^ และเก็บผลลัพธ์ลงในสแต็กโดยใช้ฟังก์ชัน push().
- ท้ายที่สุด ค่าผลลัพธ์สุดท้ายจะถูกแสดงผลโดยใช้ฟังก์ชัน pop() เพื่อดึงค่าออกจากสแต็กและแสดงผลลัพธ์ทางหน้าจอ.

```
84 int main() {
85   printf("POSTFIX CALCULATION PROGRAM\n");
86   printf("============================\n");
87   CalPostfix(postfix1);
88   getch();
89   return (0);
90 } // End Main
```

main(): นี่คือฟังก์ชันหลักที่เรียกใช้ฟังก์ชัน CalPostfix() เพื่อคำนวณค่านิพจน์ Postfix และแสดงผลลัพธ์ให้ผู้ใช้เห็นผ่านทางหน้าจอของคอมพิวเตอร์.

Run Program



สรุปผลการทดลอง

โปรแกรมนี้ใช้สแต็กในการจัดเก็บค่าตัวเลขและตัวดำเนินการเพื่อคำนวณค่าของนิพจน์ Postfix และแสดงผลลัพธ์ให้ผู้ใช้เห็นผ่านทางหน้าจอของคอมพิวเตอร์.