รายงาน

# เรื่อง โปรแกรม PUSH/POP function of Stack

จัดทำโดย

นายพงษ์พันธุ์ เลาวพงศ์
รหัสนักศึกษา 66543206019-2

เสนอ

อาจารย์ปิยพล ยืนยงสถาวร

รายงานนี้เป็นส่วนหนึ่งของวิชา
ENGCE124
โครงสร้างข้อมูลและขั้นตอนวิธี
(Data Structures and Algorithms)
หลักสูตร วศ.บ.วิศวกรรมคอมพิวเตอร์
สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา ภาคพายัพ เชียงใหม่
ภาคเรียนที่ 1 ปีการศึกษา 2567

รายงาน

# เรื่อง โปรแกรม PUSH/POP function of Stack


จัดทำโดย

นายพงษ์พันธุ์ เลาวพงศ์

รหัสนักศึกษา 66543206019-2


เสนอ

อาจารย์ปิยพล ยืนยงสถาวร

รายงานนี้เป็นส่วนหนึ่งของวิชา

ENGCE124

โครงสร้างข้อมูลและขั้นตอนวิธี

(Data Structures and Algorithms)

หลักสูตร วศ.บ.วิศวกรรมคอมพิวเตอร์


สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา ภาคพายัพ เชียงใหม่


ภาคเรียนที่ 1 ปีการศึกษา 2567

# คำนำ

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา ENGCE124 โครงสร้างข้อมูลและขั้นตอนวิธี (Data Structures and Algorithms) หลักสูตร วศ.บ.วิศวกรรมคอมพิวเตอร์ สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา ภาคพายัพ เชียงใหม่ ในระดับปริญญาตรีปีที่ 2 โดยมีจุดประสงค์ในการอธิบายโค้ดของโปรแกรม PUSH/POP function of Stack รวมถึงอธิบายหลักการทำงานของโปรแกรม PUSH/POP function of Stack และอธิบายผลลัพธ์การใช้งานโปรแกรม PUSH/POP function of Stack

ผู้จัดทำรายงานหวังว่า รายงานฉบับนี้จะเป็นประโยชน์กับผู้ที่สนใจ หรือนักศึกษาทุกท่าน ที่กำลังหาศึกษาในหัวข้อของโปรแกรม PUSH/POP function of Stack หากมีข้อแนะนำหรือข้อผิดพลาดประการใด ผู้จัดทำขอน้อมรับไว้ และขออภัยมา ณ ที่นี้

ผู้จัดทำ

นายพงษ์พันธุ์ เลาวพงศ์

วันที่ 29/07/2567

# สารบัญ

# โค้ดของโปรแกรม PUSH/POP function of Stack พร้อมคำอธิบาย

```
/*
Program create PUSH/POP function of Stack and use its.
The program will exit when status are "OVER FLOW" or "UNDER FLOW".
============================================================
*/
#include <stdio.h> //สำหรับใช้งานฟังก์ชัน printf()
#include <conio.h>//สำหรับใช้งานฟังก์ชัน getch()
#define MaxStack 6//ตั้งค่าการใช้งาน Stack สูงสุด
int stack[MaxStack]; //กำหนด Stack ข้อมูลสูงสุด 0-5
int x; //กำหนดตัวแปรชั่วคราว
int SP = 0; //กำหนดตัวแปร SP = 0 [สำหรับเป็นตัวแปรในการชี้ข้อมูลในสแตก]
char status = 'N'; //กำหนดตัวแปร status ให้มีค่าเป็น N เพื่อใช้งานในการบ่งบอกสถานะปกติ [Status = Normal]
char ch;//กำหนดตัวแปรสำหรับค่าจากคีย์บอร์ด
void push(int x) //ฟังก์ชันของ PUSH
{
    if(SP == MaxStack-1){ //เงื่อนไขตรวจสอบความจุของสแตก [ว่าสแตกเต็มหรือไม่]
        printf("!!!OVER FLOW!!!...\n");
        status='O'; //set status = OVER FLOW //ถ้าเงื่อนไขข้างต้นเป็นจริงจะกำหนดค่าสถานะเป็น o หรือสถานะ OVER FLOW
    }
    else
    {
        SP=SP+1; //ถ้าเงื่อนไขข้างต้นเป็นเท็จ ให้ค่า SP เพิ่มขึ้น 1 [ตัวแปรย้ายตำแหน่งและชี้เพิ่มขึ้นอีก 1 ช่อง]
        stack[SP]=x; //นำข้อมูลใหม่ไปใส่ในตำแหน่งที่ SP ชี้ในสแตก
    }
}
```

# โค้ดของโปรแกรม PUSH/POP function of Stack พร้อมคำอธิบาย (ต่อ)

```
int pop() //ฟังก์ชันของ POP
{
    int x;
    if (SP != 0) //เงื่อนไขตรวจสอบข้อมูลในสแตก [ข้อมูลในสแตกว่างหรือไม่]
    {
        x=stack[SP]; // ถ้าเงื่อนไขเป็นจริงนำข้อมูลออกจากสแตก
        SP--; // กำหนดให้ค่า SP ลดลง 1 [ตำแหน่งของ SP ลดลง 1 ช่อง]
        return(x); // ส่งค่าข้อมูล x ออกนอกฟังก์ชันสำหรับประมวลผล
    }
    else
    {
        printf("\n!!!UNDER FLOW!!!...\n");
        status = 'U'; //ถ้าเงื่อนไขข้างต้นเป็นจริงจะกำหนดค่าสถานะเป็น o หรือสถานะ OVER FLOW
    }
}
void ShowAllStack() //ฟังก์ชันการแสดงผล
{
    int i; //สร้างตัวแปรสำหรับการนับ
    printf(" N : %d\n ",MaxStack-1); //แสดงผลค่า N
    printf("Status : %c\n ",status); //แสดงผลค่าสถานะ [Status]
    printf("SP : %d\n",SP); //แสดงผลค่า SP
    for ( i = 1; i < MaxStack; i++ )
    {
        printf("%d:%d ",i, stack[i]); //แสดงผลข้อมูลทั้งหมดในสแตก
    }
    printf("\n-------------------------------------------------------------\n");
}
```

# โค้ดของโปรแกรม PUSH/POP function of Stack พร้อมคำอธิบาย (ต่อ)

```
int main()
{
    printf("STACK PROGRAM...\n");
    printf("==================\n");
    while (status == 'N') //เงื่อนไขถ้าหากสถานะมีค่าเป็นปกติ [Normal]
    {
        printf("[1=PUSH : 2=POP] : "); //แสดงผลเลือกเมนู push หรือ pop
        ch = getch(); //กำหนดให้ ch รอการกรอกข้อมูลผ่านคีย์บอร์ด โดยต้องกด Enter
        switch(ch) //ตรวจสอบการทำงานของ ch
           {
                  case '1' : printf("\nEnter Number : ");
                       scanf("%d", &x); //รับค่าจากคีย์บอร์ด
                       push(x); //ใช้งานฟังก์ชัน push
                       ShowAllStack(); //แสดงผลข้อมูลทั้งหมดในสแตก
                       break;
                  case '2' : x=pop(); //ใช้งานฟังก์ชัน pop
                       printf("\nData : %d\n",x); //แสดงผลข้อมูลหลังการ pop
                       ShowAllStack(); //แสดงผลข้อมูลทั้งหมดในสแตก
                       break;
           } //End SWITCH CASE
       } //End WHILE Loop
       printf("\n"); //แสดงผลบรรทัดใหม่
       return(0);
}//End MAIN Fn.
```

# หลักการทำงานของโปรแกรม PUSH/POP function of Stack

1. การประกาศและกำหนดค่าเริ่มต้น

```
#include <stdio.h> //สำหรับใช้งานฟังก์ชัน printf()

#include <conio.h>//สำหรับใช้งานฟังก์ชัน getch()

#define MaxStack 6//ตั้งค่าการใช้งาน Stack สูงสุด

int stack[MaxStack]; //กำหนด Stack ข้อมูลสูงสุด 0-5

int x; //กำหนดตัวแปรชั่วคราว

int SP = 0; //กำหนดตัวแปร SP = 0 [สำหรับเป็นตัวแปรในการชี้ข้อมูลในสแตก]

char status = 'N'; //กำหนดตัวแปร status ให้มีค่าเป็น N เพื่อใช้งานในการบ่งบอกสถานะปกติ [Status = Normal]

char ch;//กำหนดตัวแปรสำหรับค่าจากคีย์บอร์ด
```

- ส่วนนี้เป็นการรวม header files และการประกาศตัวแปรที่จำเป็นต้องใช้ในโปรแกรม
- MaxStack กำหนดขนาดสูงสุดของ stack
- stack เป็น array ที่ใช้เก็บข้อมูลใน stack
- x เป็นตัวแปรชั่วคราวสำหรับเก็บข้อมูลที่ใช้ในการ push หรือ pop
- SP เป็นตัวแปร stack pointer ที่ชี้ไปยังตำแหน่งบนสุดของ stack
- status เป็นตัวแปรที่ใช้เก็บสถานะของ stack (N: NORMAL, O: OVER FLOW, U: UNDER FLOW)
- ch เป็นตัวแปรที่ใช้เก็บค่าที่อ่านจากคีย์บอร์ด

2. ฟังก์ชัน push

```
void push(int x) //ฟังก์ชันของ PUSH
{
    if(SP == MaxStack-1){ //เงื่อนไขตรวจสอบความจุของสแตก [ว่าสแตกเต็มหรือไม่]
        printf("!!!OVER FLOW!!!...\n");
        status='O'; //ถ้าเงื่อนไขข้างต้นเป็นจริงจะกำหนดค่าสถานะเป็น o หรือสถานะ OVER FLOW
    }
    else
    {
        SP=SP+1; //ถ้าเงื่อนไขข้างต้นเป็นเท็จ ให้ค่า SP เพิ่มขึ้น 1 [ตัวแปรย้ายตำแหน่งและชี้เพิ่มขึ้นอีก 1 ช่อง]
        stack[SP]=x; //นำข้อมูลใหม่ไปใส่ในตำแหน่งที่ SP ชี้ในสแตก
    }
}
```

- ฟังก์ชันนี้ใช้ในการเพิ่มข้อมูลเข้า stack
- เช็คว่า stack เต็มหรือไม่ (SP == MaxStack-1)
    - ถ้าเต็มจะแสดงข้อความ "!!!OVER FLOW!!!" และเปลี่ยนสถานะเป็น 'O' (OVER FLOW)
    - ถ้าไม่เต็ม จะเพิ่มค่า SP ขึ้น 1 และเก็บข้อมูลใน stack ที่ตำแหน่ง SP

3. ฟังก์ชัน pop

```
int pop() //ฟังก์ชันของ POP
{
    int x;
    if (SP != 0) //เงื่อนไขตรวจสอบข้อมูลในสแตก [ข้อมูลในสแตกว่างหรือไม่]
    {
        x=stack[SP]; // ถ้าเงื่อนไขเป็นจริงนำข้อมูลออกจากสแตก
        SP--; // กำหนดให้ค่า SP ลดลง 1 [ตำแหน่งของ SP ลดลง 1 ช่อง]
        return(x); // ส่งค่าข้อมูล x ออกนอกฟังก์ชันสำหรับประมวลผล
    }
    else
    {
        printf("\n!!!UNDER FLOW!!!...\n");
        status = 'U'; //ถ้าเงื่อนไขข้างต้นเป็นจริงจะกำหนดค่าสถานะเป็น o หรือสถานะ OVER FLOW
    }
}
```

- ฟังก์ชันนี้ใช้ในการดึงข้อมูลออกจาก stack
- เช็คว่า stack ไม่ว่าง (SP != 0)
    - ถ้าไม่ว่าง จะดึงข้อมูลจาก stack ที่ตำแหน่ง SP และลดค่า SP ลง 1 จากนั้นส่งคืนข้อมูล
    - ถ้าว่าง จะแสดงข้อความ "!!!UNDER FLOW!!!" และเปลี่ยนสถานะเป็น 'U' (UNDER FLOW)

4. ฟังก์ชัน ShowAllStack

```
void ShowAllStack() //ฟังก์ชันการแสดงผล
{
    int i; //สร้างตัวแปรสำหรับการนับ
    printf(" N : %d\n ",MaxStack-1); //แสดงผลค่า N
    printf("Status : %c\n ",status); //แสดงผลค่าสถานะ [Status]
    printf("SP : %d\n",SP); //แสดงผลค่า SP
    for ( i = 1; i < MaxStack; i++ )
    {
        printf("%d:%d ",i, stack[i]); //แสดงผลข้อมูลทั้งหมดในสแตก
    }
    printf("\n-------------------------------------------------------------\n");
}
```

- ฟังก์ชันนี้ใช้ในการแสดงข้อมูลทั้งหมดใน stack
- แสดงค่าขนาดสูงสุดของ stack (N), สถานะปัจจุบัน (status), ค่า SP, และข้อมูลทั้งหมดใน stack

5. ฟังก์ชัน main

```
int main()
{
    printf("STACK PROGRAM...\n");
    printf("==================\n");
    while (status == 'N') //เงื่อนไขถ้าหากสถานะมีค่าเป็นปกติ [Normal]
    {
        printf("[1=PUSH : 2=POP] : "); //Show MENU //แสดงผลเลือกเมนู push หรือ pop
        ch = getch(); //กำหนดให้ ch รอการกรอกข้อมูลผ่านคีย์บอร์ด โดยต้องกด Enter
        switch(ch) //ตรวจสอบการทำงานของ ch
           {
                  case '1' : printf("\nEnter Number : ");
                       scanf("%d", &x); //รับค่าจากคีย์บอร์ด
                       push(x); //ใช้งานฟังก์ชัน push
                       ShowAllStack();//แสดงผลข้อมูลทั้งหมดในสแตก
                       break;
                  case '2' : x=pop(); //ใช้งานฟังก์ชัน pop
                       printf("\nData : %d\n",x); //แสดงผลข้อมูลหลังการ pop
                       ShowAllStack(); //แสดงผลข้อมูลทั้งหมดในสแตก
                       break;
           } //End SWITCH CASE
       } //End WHILE Loop
       printf("\n"); //แสดงผลบรรทัดใหม่
       return(0);
}//End MAIN Fn.
```

- ฟังก์ชันนี้เป็นฟังก์ชันหลักของโปรแกรม
- แสดงข้อความต้อนรับและแสดงเมนูการใช้งาน (PUSH หรือ POP)
- รอและอ่านค่าจากคีย์บอร์ดโดยไม่ต้องกด ENTER
- ใช้คำสั่ง switch-case ในการตรวจสอบค่าที่อ่านมา
    - ถ้ากด '1' จะให้กรอกข้อมูลและเรียกฟังก์ชัน push จากนั้นแสดงข้อมูลทั้งหมดใน stack

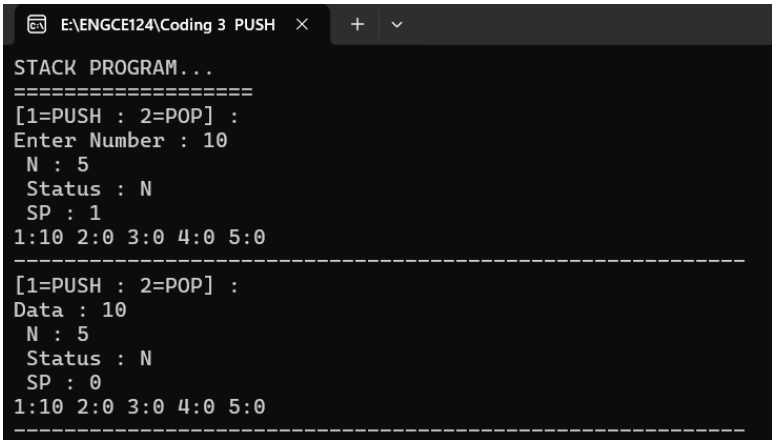
    - ถ้ากด '2' จะเรียกฟังก์ชัน pop และแสดงข้อมูลที่ดึงออกมา จากนั้นแสดงข้อมูลทั้งหมดใน stack
- วนลูปไปเรื่อย ๆ จนกว่าสถานะจะไม่เป็น 'N' (NORMAL) ซึ่งจะเป็น 'O' (OVER FLOW) หรือ 'U' (UNDER FLOW)

การทำงานโดยรวมของโปรแกรมนี้คือการรับข้อมูลจากผู้ใช้และจัดการข้อมูลใน stack ด้วยการทำงานของฟังก์ชัน push และ pop ตามลำดับ พร้อมทั้งแสดงข้อมูลใน stack ทุกครั้งที่มีการเปลี่ยนแปลง

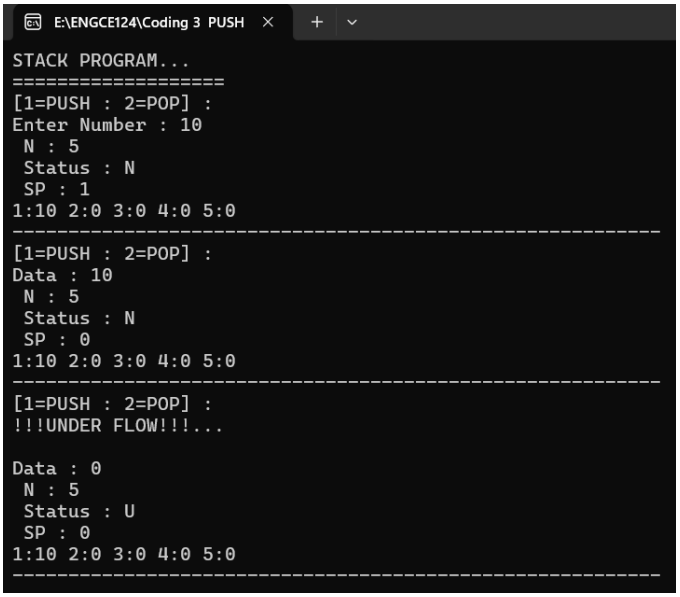
## ผลลัพธ์การใช้งานโปรแกรม PUSH/POP function of Stack

1. เมื่อเริ่มต้นโปรแกรม จะแสดงข้อความต้อนรับและเมนูให้ผู้ใช้เลือกทำงาน (PUSH หรือ POP)

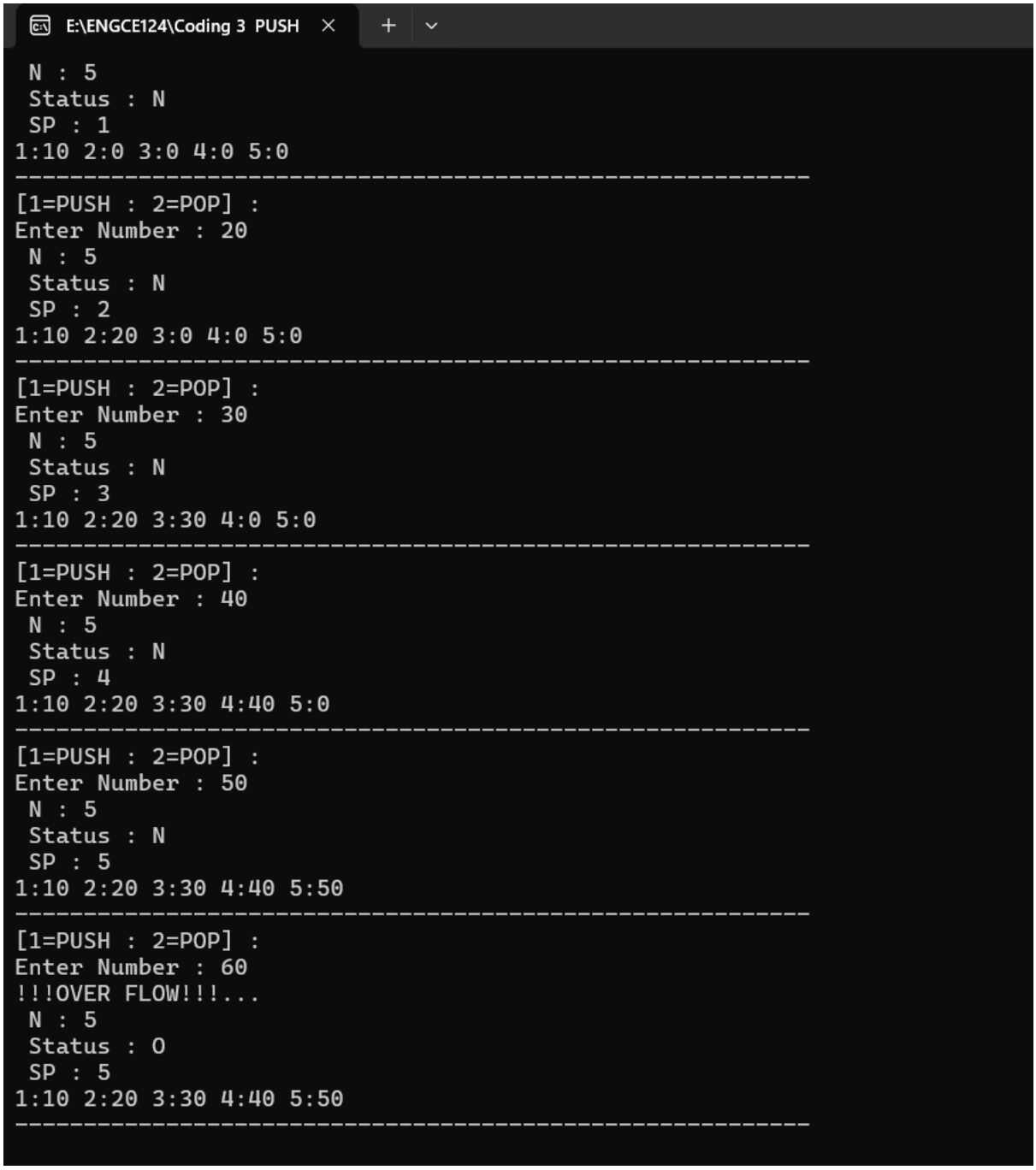
- กรณีผู้ใช้เลือก PUSH
    - ผู้ใช้กดปุ่ม 1 และโปรแกรมจะขอให้กรอกข้อมูลที่จะใส่ใน Stack
        - หลังจากผู้ใช้กรอกข้อมูล เช่น 5 และกด Enter โปรแกรมจะเรียกฟังก์ชัน push เพื่อเพิ่มข้อมูลลง Stack
        - ถ้า Stack ไม่เต็ม โปรแกรมจะเพิ่มข้อมูลและแสดงข้อมูลทั้งหมดใน Stack
- กรณีผู้ใช้เลือก POP
    - ผู้ใช้กดปุ่ม 2 และโปรแกรมจะเรียกฟังก์ชัน pop เพื่อดึงข้อมูลออกจาก Stack




2. ผลลัพธ์เมื่อผู้ใช้เลือก POP หลายครั้งจน Stack ว่าง (SP = 0) โปรแกรมจะแสดงข้อความ "UNDER FLOW" และเปลี่ยนสถานะเป็น 'U'



3. ผลลัพธ์เมื่อผู้ใช้เลือก PUSH หลายครั้งจน Stack เต็ม (SP = MaxStack - 1) โปรแกรมจะแสดงข้อความ "OVER FLOW" และเปลี่ยนสถานะเป็น 'O'




โปรแกรมนี้ทำหน้าที่ในการจัดการ Stack โดยมีฟังก์ชัน PUSH สำหรับเพิ่มข้อมูลลงใน Stack และฟังก์ชัน POP สำหรับดึงข้อมูลออกจาก Stack โดยจะแสดงสถานะ "OVER FLOW" เมื่อ Stack เต็มและสถานะ "UNDER FLOW" เมื่อ Stack ว่าง ในแต่ละขั้นตอนจะมีการแสดงข้อมูลใน Stack รวมถึงสถานะและตำแหน่งของ Stack Pointer เพื่อให้ผู้ใช้สามารถเห็นการเปลี่ยนแปลงของข้อมูลใน Stack ได้

# บรรณานุกรม

ChatGPT. ( - ). Implementation of Stack Data Structure with PUSH and POP Functions and Handling OVER FLOW and UNDER FLOW Conditions. สืบค้น 29 กรกฎาคม 2567, จาก https://chatgpt.com/