資訊工程學系
 資料結構應用
 文件編號:
 LAB4

 發佈日期:
 2022/03/21

LAB 4 以指標建構推疊資料結構

長榮大學 資訊工程學系

班級: 資工 2B

姓名:郭智榮

學號:109B30612

日期:2022/03/21

版本校定紀錄:

版本	更新紀錄	發佈日期	
0.0.0.0	初版完成	2022/03/21	

資料結構應用	文件編號:	LAB4
只有"哈特"态为	發佈日期:	2022/03/21

資訊工程學系

一. 實驗需求:

(一). 題目說明

描述:

撰寫一程式用於操作推疊(Stack),功能需包含放入資料(Push)、 移除資料(Pop)及輸出(Output)推疊資料。

輸入說明:

使用者輸入 0~3 選擇操作模式; 0 為退出 Stack; 1 為 Push; 2 為 Pop; 3 為 Output。

輸出說明:

依照使用者輸入的操作模式輸出資料。

(二). 演算法

```
定義 Stack_Size = n //用於定義 Stack 大小,變數 n 代表 Stack 大小。
int stack[Stack_Size] //創建陣列 stack, 其大小為 Stach_Size。
int input, top = -1 //input 變數儲存操作模式; top 為 stack 頂部位置-1。
int *now = stack //建立 now 儲存 stack 在記憶體的位置,方便操作使用。
void push(){
    if(top == Stack_Size-1){ //檢查 top 位置是否在 stack 的頂部。
       output 堆疊已滿 //輸出「堆疊已滿」。
    }
   else{
       output 輸入第 x 個整數: //x 代表堆疊目前有 a 個資料+1。
       (now + top + 1) = input //將使用者輸入除到 now+top+1。
       top ++; //top += 1 •
    }
void pop(){
    if(top == -1){ //檢查 top 是某在 Stack 最底部。
       output 堆疊以空 //輸出「堆疊以空」。
```

}

文件編號:LAB4發佈日期:2022/03/21

```
else{
       output 移除第 x 個整數:y //x 為 top + 1; y 為 now + top 的值。
       top --; //top -= 1 °
    }
}
void output(){
   if(top == -1){ //檢查 top 是否在 Stack 底部。
       output 堆疊以空 //輸出「堆疊以空」。
   }
   else{
       output 目前堆疊的資料: //輸出「目前堆疊的資料: 」。
       for i in range(top+1){
           output x //x 為 now + i 的值。
       }
       ouput Botton <-- Stack --> Top //輸出題目格式。
    }
}
do{
   Output === Stack Operation === //輸出題目格式。
   Output 1. Push Operation //輸出題目格式。
   Output 2. Pop Operation //輸出題目格式。
   Output 3. Printout Stack //輸出題目格式。
   Output 0. Quit //輸出題目格式。
   Output ========== //輸出題目格式。
   Output Please Enter [0~3]: //輸出題目格式。
    input = input //將使用者輸入的操作模式存到 input。
   switch(input){ //使用 switch 操作。
       case 1: //當 input == 1。
           push() //呼叫 push 功能。
              break //跳出,避免 switch 繼續執行下方的程式。
       case 2: //當 input == 2。
           pop() //呼叫 pop 功能。
           break //跳出,避免 switch 繼續執行下方的程式。
```

資訊工程學系

資料結構應用

文件編號: 發佈日期:

LAB4 2022/03/21

case 3: //當 input == 3。
 Output() //呼叫 Output 功能。
 break //跳出,避免 switch 繼續執行下方的程式。

case 0: //當 input == 0。
 break //跳出,避免 switch 繼續執行下方的程式。

default: //當 input 與 case 不相符。
 Output 請輸入 0~3 的數值 //輸出「請輸入 0~3 的數值」。
 break //跳出,避免 switch 繼續執行下方的程式。

}while(input != 0)

}

Output 程式結束,堆疊關閉 //輸出「程式結束,堆疊關閉」。

二. 完整程式碼:

```
#include<stdio.h>
#include<stdlib.h>
#define StackSize 3
int main(int argc, char *argv[]){
    int stack[StackSize];
    int input, top = -1, *now = stack;
   void push(){
       if(top == (StackSize - 1)){
           printf(" ...... Can't enter, because the stack is
full!! \n"); //此處超過 Word 大小;為方便閱讀故字體不調整。
        }
       else{
            printf("Enter No. %d integer to the stack: ", top + 2);
            scanf("%d", now + top + 1);
            top += 1;
        }
    }
   void pop(){
       if(top == -1){
           printf(" ...... Can't remove, because the stack is
empty!! \n"); //此處超過 Word 大小;為方便閱讀故字體不調整。
       else{
            printf("Remove No.%d integer from the stack : %d ", top + 1 ,
*(now + top)); //此處超過 Word 大小;為方便閱讀故字體不調整。
            top -= 1;
       }
    }
   void Output(){
       if(top == -1){
```

```
printf(" ...... Can't check, because the stack is empty!!
\n"); //此處超過 Word 大小;為方便閱讀故字體不調整。
       }
       else{
           printf("now data from the stack :");
            for(int i = 0; i < top + 1; i++){
               printf(" %d ", *(now + i));
            }
           printf("\n
                                              Botton <-- Stack -->
Top"); //此處超過 Word 大小;為方便閱讀故字體不調整。
    }
   do{
       printf("\n");
       printf("\n === Stack Operation === ");
       printf("\n 1. Push Operation ");
       printf("\n 2. Pop Operation ");
       printf("\n 3. Printout Stack ");
       printf("\n 0. Quit ");
       printf("\n ========= ");
       printf("\n Please Enter [0~3] : ");
       scanf("%d", &input);
       printf("\n\n");
       switch(input){
           case 1:
               push();
               break;
           case 2:
               pop();
               break;
           case 3:
               Output();
```

 資訊工程學系
 資料結構應用
 文件編號:
 LAB4

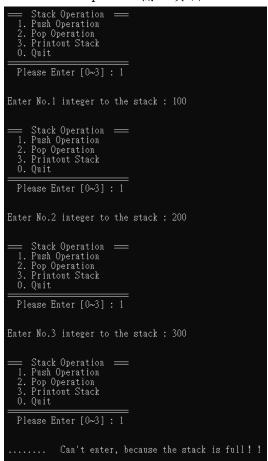
 發佈日期:
 2022/03/21

```
break;
case 0:
    break;
default:
    printf(" Only enter 0~3 \n");
}

while(input != 0);
printf("Stack off.");
return 0;
}
```

三. 輸入及輸出結果:

Step 1: 輸入資料



Step 2: 檢查堆疊資料及刪除資料

```
= Stack Operation = 1. Push Operation 2. Pop Operation 3. Printout Stack 0. Quit

Please Enter [0~3] : 3

now data from the stack : 100 200 300 Botton <-- Stack --> Top

= Stack Operation = 1. Push Operation 2. Pop Operation 3. Printout Stack 0. Quit

Please Enter [0~3] : 2

Remove No.3 integer from the stack : 300

= Stack Operation = 1. Push Operation 2. Pop Operation 3. Printout Stack 0. Quit

Please Enter [0~3] : 2

Remove No.2 integer from the stack : 200

= Stack Operation = 1. Push Operation 2. Pop Operation 3. Printout Stack 0. Quit

Please Enter [0~3] : 2

Remove No.2 integer from the stack : 200

= Stack Operation = 1. Push Operation 2. Pop Operation 3. Printout Stack 0. Quit

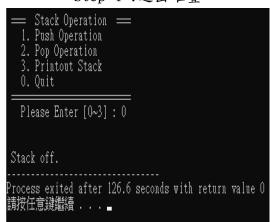
Please Enter [0~3] : 2

Remove No.1 integer from the stack : 100
```

Step 3: 刪除資料及檢查堆疊資料

Coop o . Min X Houm - PEX H
Stack Operation Push Operation Pop Operation Printout Stack Quit
Please Enter [0~3]: 2
Can't remove, because the stack is empty!!
== Stack Operation ==
1. Push Operation
2. Pop Operation 3. Printout Stack
0. Quit
Please Enter [0~3] : 3
Can't check, because the stack is empty!!

Step 4:退出堆疊



資訊工程學系	資料結構應用	文件編號:	LAB4
		發佈日期:	2022/03/21

四. 心得與討論:

本次施作的堆疊(Stack)之前在資料結構就經常提到,這次也是第一次實際操作;但在上學期的資料結構中王俊嘉教授多次以生動的方式講解,也讓我在這一次實作上能更得心應手。本次使用到的副程式撰寫,在 Java 及 Arduino 中已經大概學過語法。而副程式內的撰寫方式,在平時 Python 的一些小專題上也已經多次使用到,因此這次作業整體上遇到的困難並不多。