|  |  |
| --- | --- |
| 이름 | 2171262 박소희 |
| 주제 | Chap 04. STACK |
| 일시 | 2023 04.06 18:00 ~ 19:30 디스코드 |

이번 시간에는 chap 04을 공부하고 연습문제를 각자 풀어보기로 하였다

* 일반적인 배열 스택 프로그램(교재 예제 문제)

#include <stdio.h>

#include <stdlib.h>

#define MAX\_STACK\_SIZE 100

typedef int element; //int형을 element로 할게

element stack[MAX\_STACK\_SIZE];

int top = -1;

int is\_empty()

{

return (top == -1);

}

int is\_full()

{

return (top == (MAX\_STACK\_SIZE -1));

}

void push(element item)

{

if(is\_full()) //boolean 값

{

fprintf(stderr , "스택 포화 에러\n");

return;

}

else stack[++top] = item; //그렇지 않으면 삽입

}

element pop()

{

if(is\_empty())

{

fprintf(stderr, "스택 공백 에러\n");

exit(1);

}

else return stack[top--]; //상단을 꺼내고 인덱스 감소

}

element peek()

{

if(is\_empty())

{

fprintf(stderr, "스택 공백 에라\n");

exit(1);

}

else return stack[top];

}

int main(void)

{

push(1);

push(2);

push(3);

for(int i = 0; i < 3; i++)

{

printf("%d is out\n",pop());

}

return 0;

}

* 실행화면

텍스트, 폰트, 스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명

1 -> 2 -> 3 삽입.

3 -> 2 -> 1 제거 순서임.

STACK은 First In Last Out 구조이므로 가장 먼저 삽입된 1이

가장 나중에 제거되는 원리.

* 교재 연습문제 13번

스택을 사용하여 반복되는 문자 제거하여 출력해보는 프로그램 작성하기.

/\*배열을 활용한 스택의 구현 -> 정수를 입력받았을 때 반복되는 숫자를 제거하는 프로그램 작성\*/

#include <stdio.h>

#include <ctype.h>

int stack[100];

int top = 0;

int push(int);

void print();

int main(void)

{

int num = 0;

printf("정수를 입력하시오 : ");

scanf("%d", &num);

push(num);

print();

}

int push(int num)

{

if(top > 100)

{

printf("stack full");

exit(1);

}

printf("들어온 data : %d\n", num);

while(num>0) //num을 계속해서 나누기

{

if(stack[top]!=num%10) //stack의 top(이전 data)인덱스 요소값과 현재 num%10값이 다르다면

{

top++; //인덱스를 증가시키고

stack[top] = num%10; //stack[top]위치에 num%10(num의 일의자리)값을 대입

//stack 자료구조는 처음 들어온 data를 삽입하고 top을 증가시키는 원리이므로 num의 일의자리 숫자부터

//담기게 된다. 출력할 때는 현재 top위치부터 순서대로 출력하면 됨.

}

num /= 10; //일의 자리 숫자 제거

}

}

void print()

{

int i = 0, tmp;

printf("출력 : ");

for(tmp = top; tmp > 0; tmp--)

{

printf("%d",stack[tmp]);

if(tmp!=1) printf(",");

}

printf("\n");

}

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

* 스택을 이용하여 회문인지 아닌지를 판단하는 프로그램 작성하기

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <string.h>

int stack[100];

int top = 0;

void push(char \*);

void print();

int pop();

int main(void)

{

char s[100];

printf("문자열을 입력하시오: ");

gets(s);

push(s);

print();

if(pop()) printf("회문입니다.\n");

else printf("회문이 아닙니다.\n");

return 0;

}

void push(char \*str)

{

int i = 0;

while(str[i]!=0)

{

if(top > 100)

{

printf("stack full!!"); exit(1);

}

if(str[i]>='A' && str[i]<='Z')

{

str[i] = str[i] + 32;

}

if(str[i]>='a' && str[i] <='z')

{

top++;

stack[top] = str[i];

}

i++;

}

}

void print()

{

int tmp;

printf("출력 : ");

for(tmp = 1; tmp <= top; tmp++)

{

printf("%c",stack[tmp]);

if(tmp!=top) printf(",");

}

printf("\n");

}

int pop()

{

int val = 0;

for(int i = 1; i <= top/2; i++)

{

printf("%d %d\n",top/2, i);

if(stack[i]==stack[top-i+1]) //인덱스 값이 같으면

val++; //회문이므로 증가

}

return (val == top/2);

}

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

* 추가 과제) 재귀함수를 이용하여 계단 오르는 경우의 수 찾기

#include <stdio.h>

int climbStairs(int n){

if(n == 1)

return 1;

else if(n == 2)

return 1 + climbStairs(1);

else

return climbStairs(n - 2) + climbStairs(n - 1);

}

int main (void)

{

int n;

printf("정수 입력 : ");

scanf("%d",&n);

printf("%d가지 \n",climbStairs(n));

return 0;

}

풀이방법 : 한 번의 계단을 오르는 경우의 수는 1 또는 2 이다.

따라서 n ==1 일때, n== 2일때, 그 외 경우(2초과)일 때의 분기를 나눈다.

N이 3이상인 경우 1계단 2계단 오를 수 있으므로,

각각 n-2, n-1을 한 값을 매개변수로 던져 주어서 자기자신을 호출하면 된다.

텍스트, 폰트, 스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명