객체지향언어 프로젝트 과제

프로젝트 과제 #1 (중간고사 대체)

* 10장에서 학습한 ‘템플릿’에서 MyStack (스택)을 이용한 사칙연산 계산기를 작성해 제출하시오. ( 본 과제는 리포트 양식에 맞춰 구현 로직, 구조, 소스설명, 결과, 소스)를 도출해 제출해 주시길 바랍니다.
* 조건 1: 9장에 추상 클래스와 이에 따른 파생클래스를 이용해 사칙연산(+, -, \*, / ) 되도록 구현할 것
* 조건 2: 후위표기식으로 수식(ex, (3\*4)+(4/4) )을 표현해 MyStack을 구성해 계산할 것 (아래 링크 확인)  
  <https://repl.it/@yangkim/2020-Data-Structure#DataStrcuture/Chap.4/postfix_stack.c>  
  후위표기식: (1+2)\*7 -> 12+7\*
* 조건 3: 우선순위(괄호안에 수식 먼저 계산, \*, /의 경우 계산함) 기반의 수식 계산이 되게 할 것 (괄호는 제외하고 그냥 사칙연산 \*, /, +, - 만으로 구성해도 됨

|  |
| --- |
| // 참고용 소스 (in C)  // C program to convert infix expression to postfix  #include <stdio.h>  #include <string.h>  #include <stdlib.h>    // Stack type  struct Stack  {  int top;  unsigned capacity;  int\* array;  };    // Stack Operations  struct Stack\* createStack( unsigned capacity )  {  struct Stack\* stack = (struct Stack\*) malloc(sizeof(struct Stack));    if (!stack)  return NULL;    stack->top = -1;  stack->capacity = capacity;    stack->array = (int\*) malloc(stack->capacity \* sizeof(int));    return stack;  }  int isEmpty(struct Stack\* stack)  {  return stack->top == -1 ;  }  char peek(struct Stack\* stack)  {  return stack->array[stack->top];  }  char pop(struct Stack\* stack)  {  if (!isEmpty(stack))  return stack->array[stack->top--] ;  return '$';  }  void push(struct Stack\* stack, char op)  {  stack->array[++stack->top] = op;  }      // A utility function to check if the given character is operand  int isOperand(char ch)  {  // return (ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z');  return ( ch >= '0' && ch <= '9');  }    // A utility function to return precedence of a given operator  // Higher returned value means higher precedence  int Prec(char ch)  {  switch (ch)  {  case '+':  case '-':  return 1;    case '\*':  case '/':  return 2;    case '^':  return 3;  }  return -1;  }      // The main function that converts given infix expression  // to postfix expression.  int infixToPostfix(char\* exp)  {  int i, k;    // Create a stack of capacity equal to expression size  struct Stack\* stack = createStack(strlen(exp));  if(!stack) // See if stack was created successfully  return -1 ;    for (i = 0, k = -1; exp[i]; ++i)  {  // If the scanned character is an operand, add it to output.  if (isOperand(exp[i]))  exp[++k] = exp[i];    // If the scanned character is an ‘(‘, push it to the stack.  else if (exp[i] == '(')  push(stack, exp[i]);    // If the scanned character is an ‘)’, pop and output from the stack  // until an ‘(‘ is encountered.  else if (exp[i] == ')')  {  while (!isEmpty(stack) && peek(stack) != '(')  exp[++k] = pop(stack);  if (!isEmpty(stack) && peek(stack) != '(')  return -1; // invalid expression  else  pop(stack);  }  else // an operator is encountered  {  while (!isEmpty(stack) && Prec(exp[i]) <= Prec(peek(stack)))  exp[++k] = pop(stack);  push(stack, exp[i]);  }    }    // pop all the operators from the stack  while (!isEmpty(stack))  exp[++k] = pop(stack );    exp[++k] = '\0';  printf( "%s\n", exp );  }    // Driver program to test above functions  int main()  {  char exp[] = "a+b/b\*a-c\*d";  infixToPostfix(exp);  return 0;  } |

수식: 3+4/4\*3-2\*5 의 결과값을 과제 리포트에 결과로 제시할 것

1. 조건 1까지는 기본 과제 평가 대상입니다.
2. 조건 2와 3을 완성한 경우, 보너스 점수 반영해 과제점수 반영됩니다.