CS 211 - 2/9/18

Friday, February 9, 2018 8:21 AM

```
#define TRUE 1
#define FALSE 0

int isOpeningSymbol( char ch )
{
  if ( ch == '{' || ch == '(' || ch == '[' || ch == '<' )
     return TRUE;
  else
     return FALSE;
}

int isClosingSymbol ( char ch )
{ … }

char getMatchingSymbol ( char ch )
{
   if ( ch == '(' )  return ')';
   if ( ch == ')' )  return '(';
   if ( ch == '{' )  return '}';
   …
   return '?';    /* what to return on error? */
}

int doSymbolsMatch ( char ch1, char ch2 )
{
  char symbols[8] = { '(', '{', '[', '<', ')', '}', ']', '>' };
```

```
  int a;
  for ( a = 0 ; a < 8 ; a++ )
  {
     int b  = (a + 4) % 8;
     if ( ch1 == symbols[a] && ch2 = symbols[b] )
        return TRUE;
  }
  return FALSE;
}

/* fgets example */

char input[302];

/* to read in from standard input */
fgets ( input, 302, stdin);

/* remove the \n character */
int a = 0;
while ( input[a] != '\0' )
{
    if ( '\n' == input[a]  )
       input[a] = '\0';
     a++;
}


/* Code from lab4b.c  */
```

```
#include <stdio.h>
#include <stdlib.h>

/* typical C boolean set-up */
#define TRUE 1
#define FALSE 0

typedef struct StackStruct
{
 int*    darr;  /* pointer to dynamic array  */
 int     size;  /* amount of space allocated */
 int     inUse; /* top of stack indicator
           - counts how many values are on the stack   */
} Stack;

void init (Stack* s)
{
 s->size = 2;
 s->darr = (int*) malloc ( sizeof (int) * s->size );
 s->inUse  = 0;
}

void push (Stack* s, int val)
{
  /* check if enough space currently on stack and grow if needed */

  /* add val onto stack */
  s->darr[s->inUse] = val;
  s->inUse = s->inUse + 1;
}

int isEmpty (Stack* s)
```

```
{
  if ( s->inUse == 0)
    return TRUE;
  else
    return FALSE;
}

int top (Stack* s)
{
  return (  s->darr[s->inUse-1] );
}

void pop (Stack* s)
{
  s->inUse = s->inUse - 1;
}

void reset (Stack* s)
{
  /* how to make the stack empty? */

}

int main (int argc, char** argv)
{
 Stack st1;

 init (&st1);

 push (&st1, 20);
 push (&st1, 30);
 push (&st1, 40);
```

```
 push (&st1, 50);

 while ( isEmpty(&st1) == FALSE)
   {
    int value = top (&st1);
    printf ("The top value on the stack is %d\n", value);
    pop (&st1);
   }

 printf ("The stack is now empty\n");
}
```