**#include <stdio.h> /\* Standard input/output \*/**

**#include <string.h> /\* For string functions \*/**

**#include <ctype.h> /\* For classifying characters \*/**

**#include <stdlib.h> /\* For converting strings to numeric values \*/**

**#include <math.h> /\* For power() function \*/**

**void main()**

**{**

**char input[256]; /\* Input expression \*/**

**char number\_string[30]; /\* Stores a number string from input \*/**

**char op = 0; /\* Stores an operator \*/**

**unsigned int index = 0; /\* Index of the current a character in input \*/**

**unsigned int to = 0; /\* To index for copying input to itself \*/**

**size\_t input\_length = 0; /\* Length of the string in input \*/**

**unsigned int number\_length = 0; /\* Length of the string in number\_string \*/**

**double result = 0.0; /\* The result of an operation \*/**

**double number = 0.0; /\* Stores the value of number\_string \*/**

**printf("\nTo use this calculator, enter any expression with " " or without spaces");**

**printf("\nAn expression may include the operators:");**

**printf("\n +, -, \*, /, %%, or ^(raise to a power).");**

**printf("\nUse = at the beginning of a line to operate on ");**

**printf("\nthe result of the previous calculation.");**

**printf("\nUse quit by itself to stop the calculator.\n\n");**

**/\* The main calculator loop \*/**

**while(strcmp(gets(input), "QUIT") != 0)**

**{**

**input\_length = strlen(input); /\* Get the input string length \*/**

**/\* Remove all spaces from the input by copy the string to itself \*/**

**/\* including the string terminating character \*/**

**for(to = 0, index = 0 ; index<=input\_length ; index++)**

**if(\*(input+index) != ' ') /\* If it is not a space \*/**

**\*(input+to++) = \*(input+index); /\* Copy the character \*/**

**input\_length = strlen(input); /\* Get the new string length \*/**

**index = 0; /\* Start at the first character \*/**

**if(input[index]== '=') /\* Is there =? \*/**

**index++; /\* Yes so skip over it \*/**

**else**

**{ /\* No - look for the left operand \*/**

**/\* Look for a number that is the left operand for the 1st operator \*/**

**/\* Check for sign and copy it \*/**

**number\_length = 0; /\* Initialize length \*/**

**if(input[index]=='+' || input[index]=='-') /\* Is it + or -? \*/**

**\*(number\_string+number\_length++) = \*(input+index++); /\* Yes so copy it \*/**

**/\* Copy all following digits \*/**

**for( ; isdigit(\*(input+index)) ; index++) /\* Is it a digit? \*/**

**\*(number\_string+number\_length++) = \*(input+index); /\* Yes - Copy it \*/**

**/\* copy any fractional part \*/**

**if(\*(input+index)=='.') /\* Is it decimal point? \*/**

**{ /\* Yes so copy the decimal point and the following digits \*/**

**\*(number\_string+number\_length++) = \*(input+index++); /\* Copy point \*/**

**for( ; isdigit(\*(input+index)) ; index++) /\* For each digit \*/**

**\*(number\_string+number\_length++) = \*(input+index); /\* copy it \*/**

**}**

**\*(number\_string+number\_length) = '\0'; /\* Append string terminator \*/**

**/\* If we have a left operand, the length of number\_string \*/**

**/\* will be > 0. In this case convert to a double so we \*/**

**/\* can use it in the calculation \*/**

**if(number\_length>0)**

**result = atof(number\_string); /\* Store first number as result \*/**

**}**

**/\* Now look for 'op number' combinations \*/**

**for(;index < input\_length;)**

**{**

**op = \*(input+index++); /\* Get the operator \*/**

**/\* Copy the next operand and store it in number \*/**

**number\_length = 0; /\* Initialize the length \*/**

**/\* Check for sign and copy it \*/**

**if(input[index]=='+' || input[index]=='-') /\* Is it + or -? \*/**

**\*(number\_string+number\_length++) = \*(input+index++); /\* Yes - copy it. \*/**

**/\* Copy all following digits \*/**

**for( ; isdigit(\*(input+index)) ; index++) /\* For each digit \*/**

**\*(number\_string+number\_length++) = \*(input+index); /\* copy it. \*/**

**/\* copy any fractional part \*/**

**if(\*(input+index)=='.') /\* Is is a decimal point? \*/**

**{ /\* Copy the decimal point and the following digits \*/**

**/\* Copy point \*/**

**\*(number\_string+number\_length++) = \*(input+index++);**

**for( ; isdigit(\*(input+index)) ; index++) /\* For each digit \*/**

**\*(number\_string+number\_length++) = \*(input+index); /\* copy it. \*/**

**}**

**\*(number\_string+number\_length) = '\0'; /\* terminate string \*/**

**/\* Convert to a double so we can use it in the calculation \*/**

**number = atof(number\_string);**

**/\* Execute operation, as 'result op= number' \*/**

**switch(op)**

**{**

**case '+': /\* Addition \*/**

**result += number;**

**break;**

**case '-': /\* Subtraction \*/**

**result -= number;**

**break;**

**case '\*': /\* Multiplication \*/**

**result \*= number;**

**break;**

**case '/': /\* Division \*/**

**/\* Check second operand for zero \*/**

**if(number == 0)**

**printf("\n\n\aDivision by zero error!\n");**

**else**

**result /= number;**

**break;**

**case '%': /\* Modulus operator - remainder \*/**

**/\* Check second operand for zero \*/**

**if((long)number == 0)**

**printf("\n\n\aDivision by zero error!\n");**

**else**

**result = (double)((long)result % (long)number);**

**break;**

**case '^': /\* Raise to a power \*/**

**result = pow(result, number);**

**break;**

**default: /\* Invalid operation or bad input \*/**

**printf("\n\n\aIllegal operation!\n");**

**}**

**}**

**printf("= %f\n", result); /\* Output the result \*/**

**}**

**}**