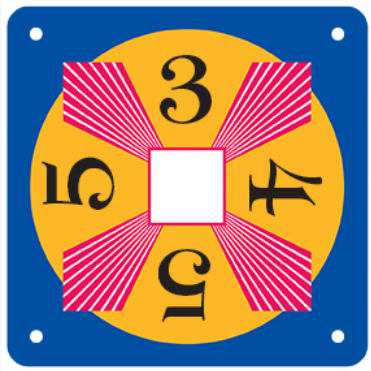
Program 3: TwentyFour Part Two

CS 211, UIC Spring 2023

Description

24 is a math game, in which the player is presented with 4 numbers and they must determine a way to combine those numbers using mathematical operations to get 24 as the result. Each number must be used, and each number can be used only once.

As an example, if the user is presented with the card shown below:



they may come up with the following solution (which may not be the only one!):
5 x 5 = 25
4 – 3 = 1
25 – 1 = 24
See https://www.24game.com/t-about-howtoplay.aspx for more information.

In Program 1, you developed a C program that allows the user to play a simplified version of this game, where calculations were done in order. For this program, you will allow the user to enter a **full expression** for their solution, which your program will then evaluate. This means that if the user is presented with the example above, they may type something like `(5 * 5) - (4 - 3)` for their solution, and your program will evaluate it as shown above!

The four numbers presented in each puzzle are all single digits (numbers 1-9). The user does not need to use all four mathematical operations.

Running the Program

The program begins by displaying a message to introduce the game, then prompts the user to select the difficulty level of the game, with the default difficulty level as easy.

```
Welcome to the game of TwentyFour Part Two!
Use each of the four numbers shown exactly once,
combining them somehow with the basic mathematical operators (+,-,*,/)
to yield the value twenty-four.
Choose your difficulty level: E for easy, M for medium, and H for hard (default is easy).
Your choice --> E
```

You may assume that the user will only type one character for the difficulty level. Capitalization does matter. If they do not enter 'E', 'M', or 'H', the difficulty level should be set to easy.

Selecting one of these corresponds to a file with puzzles of that difficulty: **`easy.txt`** contains the easy puzzles, **`medium.txt`** those of medium difficulty, and **`hard.txt`** those of hard difficulty.

Once the user selects the difficulty level, the program will select a random puzzle of that difficulty level and present it to the user.

```
Welcome to the game of TwentyFour Part Two!
Use each of the four numbers shown exactly once,
combining them somehow with the basic mathematical operators (+,-,*,/)
to yield the value twenty-four.
Choose your difficulty level: E for easy, M for medium, and H for hard (default is easy).
Your choice --> E
The numbers to use are: 4, 4, 8, 8.
```

Random number generators are not really random. When developing your program it can be helpful to always have it make the same choices you run the program. To make this happen you need to seed the random number generator with a fixed value. You should only seed the random number generator once in your program, typically near the beginning of main(). That statement will look like: `srand( 1);` **Be sure that srand(1) executes only once at the beginning of the program when you submit it!** Otherwise, your output will not match the expected output.

(If you would like different numbers each time, you may use srand(time(0));)

To get a random number, you will likely want to have something like the following:

```
rand() % numberOfPossiblePuzzles
```

Once the user has been presented with the puzzle of the four numbers, they can play the game! Prompt them to enter their solution.

```
Welcome to the game of TwentyFour Part Two!
Use each of the four numbers shown exactly once,
combining them somehow with the basic mathematical operators (+,-,*,/)
to yield the value twenty-four.
Choose your difficulty level: E for easy, M for medium, and H for hard (default is easy).
Your choice --> E
The numbers to use are: 4, 4, 8, 8.
Enter your solution: 8*4*8*4
8 * 4 = 32.
32 * 8 = 256.
256 * 4 = 1024.
Sorry. Your solution did not evaluate to 24.
```

Evaluating the Expression

When the user enters their expression, there are a couple things to check:

- They should not have entered invalid symbols such as '^', '#', etc. Note that parentheses are allowed because they may indicate the order of operations in the solution. Spaces are also allowed.
- The user must use all four numbers in their solution, and use each one only once.

If either of these are not the case, your program should print an error message and allow them to try again with a new puzzle. The check for invalid symbols takes precedence over the use of the four numbers.

```
Welcome to the game of TwentyFour Part Two!
Use each of the four numbers shown exactly once,
combining them somehow with the basic mathematical operators (+,-,*,/)
to yield the value twenty-four.
Choose your difficulty level: E for easy, M for medium, and H for hard (default is easy).
Your choice --> E
The numbers to use are: 4, 4, 8, 8.
Enter your solution: 4 ^ 8
Error! Invalid symbol entered. Please try again.

The numbers to use are: 1, 3, 3, 4.
Enter your solution: (1 + 3 + 4) * 4
Error! You must use all four numbers, and use each one only once. Please try again.

The numbers to use are: 5, 5, 7, 7.
Enter your solution: 5 + 5 - 7 % 7
Error! Invalid symbol entered. Please try again.

The numbers to use are: 4, 4, 8, 8.
Enter your solution: 4 + 4 + 8 + 8
4 + 4 = 8.
8 + 8 = 16.
16 + 8 = 24.
Well done! You are a math genius.
```

Evaluating an expression like 4 + 4 + 8 + 8, as shown in the image above, is not complicated. However, your program should be able to evaluate expressions such as the following:

- ((4 + 8) / 4) * 8 = 24
- (8 + 4) * (8 / 4) = 24
- 8 + 4 * 4 + 8 = 32

To be able to do this, your program must make use of at least one stack, which uses a linked list for its implementation and which you write on your own. Failure to do so will result in a grade of zero.

As you are implementing this, keep in mind that the order matters! When presented with an expression like 8 + 4 * 4 + 8, the steps of calculation should be the following:
4 * 4 = 16.
8 + 16 = 24.
24 + 8 = 32.
Thus, the expression 8 + 4 * 4 + 8 is calculated as 8 + (4 * 4) + 8.

After performing the calculation, the program will print “Well done! You are a math genius.” if the result is indeed 24. Otherwise, the program will print “Sorry. Your solution did not evaluate to 24.”

Once the user is done with a puzzle, the program presents the following three menu options:

```
Enter:  1 to play again,
        2 to change the difficulty level and then play again, or
        3 to exit the program.
Your choice -->
```

If the user selects 1, they are presented with another random puzzle of the same difficulty level.

If the user selects 2, they are asked again to select a difficulty level and a random puzzle of the new difficulty level is presented.

If the user selects 3, the program stops executing.

```
Enter:  1 to play again,
        2 to change the difficulty level and then play again, or
        3 to exit the program.
Your choice --> 3

Thanks for playing!
Exiting...
```

Sample output is provided at the end of this document to see what this looks like.

Submission

Submit your code to Gradescope. You only need to submit your .c file. The autograder will run automatically, and the resulting credit will be your execution points total.

By default, your *last (i.e. most recent)* submission on Gradescope is what will be graded. If you would like us to grade an earlier version of your program, use *the Activate button* to select it from your Submission History. This must be done *before* the deadline. Only one version of your program can be graded for both execution and style points.

Sample Output

```
Welcome to the game of TwentyFour Part Two!
Use each of the four numbers shown exactly once,
combining them somehow with the basic mathematical operators (+,-,*,/)
to yield the value twenty-four.
Choose your difficulty level: E for easy, M for medium, and H for hard (default is easy).
Your choice --> J
The numbers to use are: 4, 4, 8, 8.
Enter your solution: (4 * 8) * 8 *4
4 * 8 = 32.
32 * 8 = 256.
256 * 4 = 1024.
Sorry. Your solution did not evaluate to 24.

Enter:  1 to play again,
        2 to change the difficulty level and then play again, or
        3 to exit the program.
Your choice --> 1
The numbers to use are: 1, 3, 3, 4.
Enter your solution: (3 + 1 * 3) *4
1 * 3 = 3.
3 + 3 = 6.
6 * 4 = 24.
Well done! You are a math genius.

Enter:  1 to play again,
        2 to change the difficulty level and then play again, or
        3 to exit the program.
Your choice --> 2
Choose your difficulty level: E for easy, M for medium, and H for hard (default is easy).
Your choice --> H
The numbers to use are: 2, 3, 5, 7.
Enter your solution: 2+3 - 5 %7
Error! Invalid symbol entered. Please try again.

The numbers to use are: 2, 4, 7, 9.
Enter your solution: 2 * 4 * 3
Error! You must use all four numbers, and use each one only once. Please try again.

The numbers to use are: 2, 2, 6, 9.
Enter your solution: 2 * ((6 / 2) + 9)
```

```
6 / 2 = 3.
3 + 9 = 12.
2 * 12 = 24.
Well done! You are a math genius.

Enter:  1 to play again,
        2 to change the difficulty level and then play again, or
        3 to exit the program.
Your choice --> 3

Thanks for playing!
Exiting...
```