Exercise 2

Take a look at the file `arrays.c`. The program should print powers of two for values of exponent ranging from 0 to 9 and the sums if the first N powers of two for values of N ranging from 0 to 9.

When compiles and executed, the program produces strange results. Some of the values seem to be correct, but others are clearly wrong.

Here is a sample run of this program:

```
 $ gcc -Wall -g arrays.c -o arrays
$ ./arrays
2^0 =    1  1^0 + ... + 1^0 =     1
2^1 =    2  1^0 + ... + 1^1 =     3
2^2 =    4  1^0 + ... + 1^2 =     7
2^3 =    8  1^0 + ... + 1^3 =    15
2^4 =   16  1^0 + ... + 1^4 =    31
2^5 =   32  1^0 + ... + 1^5 =    63
2^6 =   64  1^0 + ... + 1^6 =  1041
2^7 =  128  1^0 + ... + 1^7 =     0
2^8 =    1  1^0 + ... + 1^8 = 540565042
2^9 =    3  1^0 + ... + 1^9 = 538976317
```

Use the `gdb` debugger to learn what is going on.

Here are some `gdb` instructions that may come in handy:

`display VAR_NAME` - same as print, but the value of the VAR_NAME is shown after every step through the program (not only when the instruction is executed)

`print/display &VAR_NAME` - shows the address of a variable named VAR_NAME

Create a file called "answers_ex2". You will submit this file on NYU Classes. Answer the questions below in that file. You will need to run the `arrays` program in `gdb` debugger to answer most of these questions.

1. What are the memory addresses at which both arrays are stored? Specify the instruction that you used to figure that out.

   Hint: The address you are after is the address of the zero'th index in the array (of course, there are many ways of printing that value).

2. Once the loop on lines 18-22 completes, what are the values saved in the `vals` array? Specify the `gdb` instructions that you used. List all the values in the array.

   Hint: you can set a breakpoint on line 23 and just let the program continue untill it hist that breakpoint - this way you do not have to manually step through every iteration of the loop.

3. Once the loop on lines 26-30 completes, what are the values saved in the `vals` array? what are the

values saved in the `partial_sums` array? Specify the `gdb` instructions that you used. List the values in both arrays.

Are the numbers stored in the two arrays what they supposed to be?

4. Go back to your answer to question 1. What is the difference between the two memory addresses (in decimal)? Does this make sense?

5. Run the program to line 45 and examine the values of the `partial_sums` array? Are those the values that are printed when the program is actually executed? Show the values actually stored in the array and the values that are printed when the program is executed.

Even if you figured out already what the problem with the code is, continue with the following questions. They show you some tricks that may come in handy in the future.

6. You might have discovered that printing the content of an array is rather tedious during the debugging process. There are some `gdb` instructions that may help with that.

    - Quit the debugger and start it with the same (unfixed) program again.
    - Set the breakpoints to lines 24, 34 and 45.
    - Run the program to the first breakpoint on line 24.
    - Execute the following instruction:

      `x/10dw vals`

      The `x` instruction allows you to print the content of memory. The followup characters specify what and how to print: - `10` tells it to print 10 values (this is the size of array `vals` - `d` tells it to print signed integers in decimal format - `w` tells it to print 4 byte chunks `vals` is the memory address that is the starting point of the block of values that should be printed
    - Run the program to the second breakpoint on line 34.
    - Execute:

      ```
      x/10dw vals
      x/10dw partial_sums
      ```

      Do you see the overlap in memory that those two arrays occupy?
    - Run the program to the third breakpoint on line 45.
    - Execute:

      ```
      x/10dw partial_sums
      ```
    - Use the `next` instruction to execute the rest of the program one line at a time (this means you

will step through the last loop one line at a time). After each line, display the content of `partial_sums` array. At what point does the content of the array change? Try to explain why it happens.