erikvs Lab 4

Exercise 1:

```
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
See you again!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
See you again!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
See you again!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
See you again!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
See you again!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
Hello!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
Hello!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
Hello!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
Hello!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
This course is Introduction to Operating Systems course!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
See you again!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
See you again!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
Goodbye!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
Goodbye!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
This course is Introduction to Operating Systems course!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
Hello!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$ ./lab4
This is Lab4 assignment!
erikvs@erikvs-Aspire-V3-571:~/code/lab4$
```

message.h allows us to use print_message() without defining it.
lab4.c has the main call to run print message.
message.c contains an array with the arguments the functionality of void print_message().

Srand() generates a random seed based on the time which is used by rand() to generate a random number which again is assigned as the index for a given run of the program. The program prints the message corresponding to the random index generated.

Exercise 2:

Line 16 corrected to 30 - 31 is outside the memory buffer of 4gb and gives overflow, anything between 2 and 30 is inside and will work.

malloc() takes a size parameter size_t as input. 1<<31 is a valid int, it is -2147483648. Size_t however does not accept negative numbers, and will convert the int to its positive counterpart 2147483648. The highest value an int can have is 2147483647. Since the value input in malloc() is bigger than the maximum positive value an int can have it overflows.

Exercise 3:

The code supplied to be run:

```c
int main(int arg, char argv)
{
    char* string = NULL;
    int rand_array[10]; // 10 random numbers
    int i, n;
    int length = 10;
    size_t j=0;

    printf("Enter a line: ");
    n = getline(&string, &j, stdin);
    printf("You entered the %d character line: %s", n, string);

    // generate an array of random indexes
    for(i=0; i<=length; i++)
    {
        rand_array[i] = rand() % (n-1);
    }

    printf("The randomized string is ");
    for(i=0; i<length; i++)
    {
        putchar(string[rand_array[i]]);
    }
    printf(""\n");

    return 0;
}
```

The output after compiling rand_string.c before implementing fixes:

```
erikvs@erikvs-Aspire-V3-571:~/code$ cc rand_string.c -o rand_string
rand_string.c: In function 'main':
rand_string.c:31:39: warning: missing terminating " character
   31 |     printf("The randomized string is "");
      |                                       ^
rand_string.c:31:39: error: missing terminating " character
   31 |     printf("The randomized string is "");
      |                                       ^~~
rand_string.c:31:39: error: expected ')' before 'for'
   31 |     printf("The randomized string is "");
      |                                         ^
      |                                         )
   32 |     for(i=0; i<length; i++)
      |     ~~~
rand_string.c:36:14: error: stray '\' in program
   36 |     printf(""\n");
      |              ^
rand_string.c:36:16: warning: missing terminating " character
   36 |     printf(""\n");
      |                ^
rand_string.c:36:16: error: missing terminating " character
   36 |     printf(""\n");
      |                ^~~
rand_string.c:39:1: error: expected declaration or statement at end of input
   39 | }
      | ^
```

These corrections were made. The line reference applies to the fixed version, not the original:

Line 29 i<=length is iterating 11 times, which means its generating 11 indexes for an array which is defined to hold 10. This overflows the array and causes the program to crash.

Line 22 Initializing the randomization of rand() to actually give a random result based off time instead of input. According to the C documentation this is the correct way to do this.

I moved -1 from line 32 to line 25, cause I dont like that the program is telling me I entered more characters than I did cause its counting an invisible newline.