Mandatory assignment 5 – Lesson 9+10

Hand in assignment 5 on LEARN.

Upload only source files and .pdf files.

All files must include:

- Assign number. Ex: Exercise 7.2 GuessANumber
- Date
- Name and student number

Lesson 9

Solve the following exercises using a debugger. You can use remote debugging with Visual Studio or similar IDE or you can use native gdb in Linux. In both cases you will use the gdb debugger.

Exercise 9.1 Count sixes

Run the following C program in the debugger.
Count how many times the variable n gets the value six.
Do not use printf().

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

int main() {
                srand(time(0));              // Intialize random number generator
        int n;

        for (int i = 0; i < 10; i++) {
                n = rand() % 6 + 1;          // Generate random number in [1,6]
        }
}
```

Do <u>not</u> hand in this exercise.

Exercise 9.2 Diceman

This program simulates the Dice man. Each morning he rolls a dice to decide what he is going to do.

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

int rollDice() {
        int n = rand() % 6 + 1;       // Generate random number in [1,6]
        return n;
}

int main() {
        srand(time(0));     // Intialize random number generator
        char action[100];

        int dice = rollDice();
        if (dice == 1) {
                strcpy(action, "Breakfast");
        }
        if (dice == 2) {
                strcpy(action, "Study");
        }
        if (dice == 3) {
                strcpy(action, "Swim");
        }
        if (dice == 4) {
                strcpy(action, "Go fishing");
        }
        if (dice == 5) {
                strcpy(action, "Call mom");
        }
        if (dice == 6) {
                strcpy(action, "Back to bed");
        }
        printf("%s\n", action);
        return 0;
}
```

The program is not optimal. Run it in the debugger and find out how it can be improved.

Hand in your improved version of the C program.

62558 Linix and C-programming

Exercise 9.3 Stack Trace

Make a program that can create the following stack trace:

Figure 1: Stack trace from Visual Studio

Figure 2: Stack trace from native gdb

Hand in your C program.

Lesson 10 Factorial

Solve the following exercises using a debugger. You can use remote debugging with Visual Studio or similar IDE or you can use native gdb in Linux. In both cases you will use the gdb debugger.

Exercise 10.1

The factorial of a number n is defined as follows: n!=n*(n-1)*...*2*1

The following C program computes the factorial, but it gives a wrong result. Run it with a small positive integer as input. Now, run it in the debugger and find the error.

```
#include <stdio.h>


unsigned long factorial(int n) {
   unsigned long f;
   int i;
   for (i = 1; i < n; i++) {
       f = f*i;
   }
   return f;
}

int main() {
    int k;
    printf("Enter a positive integer: ");
    scanf("%u", &k);
    unsigned long fk = factorial(k);
    printf("The value of %u factorial is %lu\n\n", k, fk);
}
```

Hand in the correct C program.