Operating Systems – spring 2023
Tutorial-Assignment 1

Instructor: Hans P. Reiser

Submission Deadline: Monday, January 16, 2023 – 23:59

A new assignment will be published every week. It must be completed before its submission deadline (late policy for programming assignments: up to two days, 10% penalty/day)

Lab Exercisess are theory and programming exercises discussed in the lab class. They are not graded, but should help you solve the graded questions and prepare for the final exam. Make sure to read and think about possible solutions before the lab class.

T-Questions are theory homework assignments and need to be answered directly on Canvas (quizz).

P-Questions are programming assignments. Download the provided template from Canvas. Do not fiddle with the compiler flags. Submission instructions can be found in the introductory section below.

In this week's assignment you will revist some basics from the first lecture, and get familiar with the C programming language.

1 Lab Exercises

Question 1.1: Basics

a. Enumerate the major tasks of an operating system.

Question 1.2: Secure Remote Shell (SSH)

a. Find out how to remotely access a Linux computer using SSH. You should be familiar with doing so based on experience from your Tölvuhögun (computer architecture) classes. You will have to (1) interactively access a remote terminal and (2) transfer files from and to a remote system.

b. Based on the previous part, create a remote login session on skel.ru.is and look at the file "quote.txt" in the "/tmp" folder using the commands "cat" and/or "less"

c. Also, transfer the file "whatisthis.jpg" in the "/tmp" folder on skel.ru.is to your own computer and look at the picture.

Question 1.3: Simple C program

For programming assignments, we provide templates that you have to extend with functionality according to the respective question. You can download the templates for programming assignments from Canvas. In this tutorial we use a template with a structure similar to what you will use for the programming homework assignments.

a. Get the template *tut01-template* for this programming question from Canvas and unpack it. Can you explain what the different files are good for? Which files are header files, and what are they used for?

b. Implement the function `greet` in *greet.c*. It takes an integer argument but does not provide a return value. The function shall print *Hello World!* as often as specified by the integer argument, each in a separate line. In addition, the printed lines shall be prefixed with a consecutive line counter (starting at 1). Compile the provided template with `make`.

c. Implement the function `countchr` in *countchr.c*. It returns the number of occurrences of a character in an ASCII string, both supplied by the caller. The string is represented as a pointer to a contiguous sequence of `char`s in memory. The string is terminated with a null (0) character.
Example: `countchr("Hello world",'o')` should return the value 2.

Question 1.4: Pointers and Structs

a. Explain the concept of pointers. What do the `&` and `*` operators do when working with pointers?

b. What are the types of the following variables. Why can this code formatting be misleading?

```
int* a, b;
```

c. Consider an array of `int`s in memory. How can you use the following pointers to access the fourth element? Both point at the beginning of the array.

```
int *ip;
void *vp;
```

2 T-Questions

T-Question 1.1: Basics

a. Which of the following statements about operating systems are correct? Select all correct definitions / statements! **3 T-pt**

true	false	
☐	☐	An operating system is a program that allows a user to perform specific tasks on a computer, such as creating documents and playing games.
☐	☐	An operating system is a layer between the hardware and the applications, and it aims at making hardware easier to use for applications.
☐	☐	An operating system is a type of hardware that connects a computer to the internet.
☐	☐	An operating system is a software program that is used to delete viruses and malware from a computer.
☐	☐	An operating system virtualizes physical resources (such as CPU and main memory) and makes virtual versions of these resources available to running applications.
☐	☐	An operating system provides security-relevant mechanisms such as isolation between processes.

b. Why is abstraction a central task of an operating system? **2 T-pt**

c. Log in to `skel.ru.is` using SSH with your RU account, and run the program "`genoutput`", which can be found in the folder "`/home/sty23/bin`". What is the program's output? **5 T-pt**

3 P-Questions

About the Programming Assignments

The following introductory words outline our expectations of your work and the requirements your solutions have to fit.

Write Readable Code

In your programming assignments, you are expected to write well-documented, readable code. There are a variety of reasons to strive for clear and readable code: Code that is understandable to others is a requirement for any real-world programmer, not to mention the fact that, after enough time, you will be in the shoes of one of the others when attempting to understand what you wrote in the past. Finally, clear, concise, well-commented code makes it easier to review your assignment! (This is especially important if you cannot get the assignment running. If you cannot figure out what is going on, how do you expect us to do it?)
There is no single right way to organize and document your code. It is not our intent to dictate a particular coding style for this class. The best way to learn about writing readable code is to read other people's code.
Here are some general tips for writing better code:

- Split large functions. If a function spans multiple pages, it is probably too long.
- Group related items together, whether they are variable declarations, lines of code, or functions.
- Use descriptive names for variables and procedures. Be consistent with this throughout the program.
- Comments should describe the programmer's intent, not the actual mechanics of the code. A comment which says "Find a free disk block" is much more informative than one that says "Find first non-zero element of array".

Write Compilable And Executable Code

Obscure code is bad, but uncompilable code is even worse. To increase your coding awareness, we expect you to use the GNU C compiler with some restrictions on warning-behavior as written in the makefiles. Do not change these flags! **These settings will cause the compiler to abort if there is some warning. Make sure to fix those warnings before submitting**.

As a reference platform, we will use Linux (Debian 11) on Intel x86-64 hardware. Your code needs to compile on that platform (or on skel.ru.is, which uses Red Hat Enterprise Linux 8.6).

If you are unable to write a fully working solution, at least make sure that your partial solution does compile, even though it might not produce the correct result. Document your intents and problems as comments in the source file to give your tutor a head start in understanding your code.

Groups

We assume that you will complete the programming assignments in groups of two students (single student submissions are accepted). Please feel free to discuss your solutions with other groups, but **do not share code.**

Templates And Stubs

You will find templates for all programming assignments in Canvas. Unzip them with the command `unzip assignmentXX-templates.zip`. The archives contain a directory for each individual task, wherein you can find several files:

<**task name**>**.h** A header file defining the function prototypes as listed in the assignment's description. You should not modify this.

<**task name**>**.c** Put your solution in here. This is the only file you will upload when submitting your solution.

main.c Contains the entry point in the resulting program. While we provide trivial test cases, you should write your own test code here.

Makefile Call `make` to build your sources.

These templates should ease your work as well as ours, so don't change anything unless explicitly allowed.

Assignment Submission

To submit your solution, please upload your <**task name**>**.c** file in the online Canvas assignment in the respective question. All other files are not part of your solution and should not be uploaded. They will be ignored.

P-Question 1.1: Printf

Download the template **p1** for this assignment from Canvas. You may only modify and upload the file `print.c`.

a. Write a function that prints a 64-bit signed integer and a null-terminated ASCII string with a single call to `printf`. The number and string should be separated by a whitespace. Each call to your function should print the output to a new line. For the integer use the platform-independent format from `inttypes.h`. **3 P-pt**

```
void print_line(int64_t number, char *string);
```

P-Question 1.2: String to Integer Conversion

Download the template **p2** for this assignment from Canvas. You may only modify and upload the file `parseint.c`.

a. Write a function that converts a single decimal digit in a char to an integer. You may use neither a library function nor a lookup table for this task. Return -1 if the given char is not a valid decimal digit. **2 P-pt**

```
int parseDecimalChar(char c);
```

b. Write a function that converts from an octal or decimal string to an integer. Reuse your function from above, but do not use any library function. Your function should recognize octal numbers through a leading zero. You may assume that the resulting integers fit into an int. Return -1 if the given string is not a valid octal or decimal number. **5 P-pt**

```
int parseInt(char *string);
```

Total:
10 T-pt
10 P-pt