HOMEWORK #1

SUBMISSION DUE DATE: 28/11/2019 <u>23:00</u>

INTRODUCTION

The purpose of this assignment is to get familiar with Linux and development environments. The assignment includes the following:

1- Remotely connect to the Nova server and get familiar with basic shell commands.
2- Create your first C programs.
3- Compiling, debugging and basic use of makefile.
4- Checking your code according to the guidelines given in class.

An executable implementing this assignment is provided to you, and your submission will be graded accordingly by matching your code's outputs on various inputs against the executable's outputs on the same inputs. The goal of this assignment is to reach a perfect match!

DEVELOPMENT ENVIRONMENT

Students may work in any development environment they like. However, we encourage you to use eclipse as your development tool (we will not support issues regarding other IDEs or operating systems other than Linux or Windows). Your submission will be automatically checked by a script, so your code should run properly on **Nova** - the faculty server. You can use the computers in the PC farm or remotely connect to the server as described below.

FILE TRANSFER

If you need to transfer files from your personal computer to Nova, you can use WinSCP. To connect you will need to enter your personal authentications. See the example below.
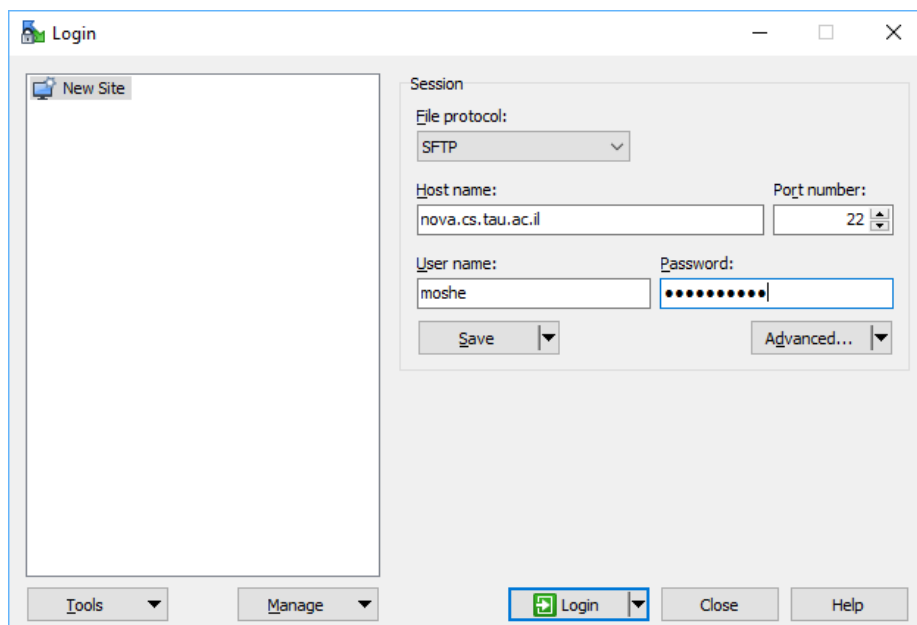
Before you press Login, you should define the tunnel by clicking **Advanced** and setting *gate.tau.ac.il* as the tunnel, as seen below:
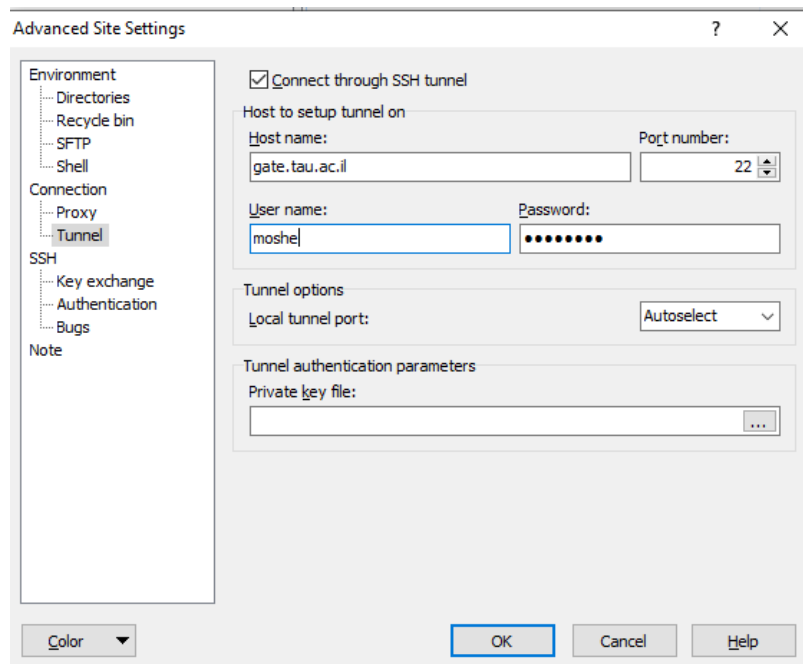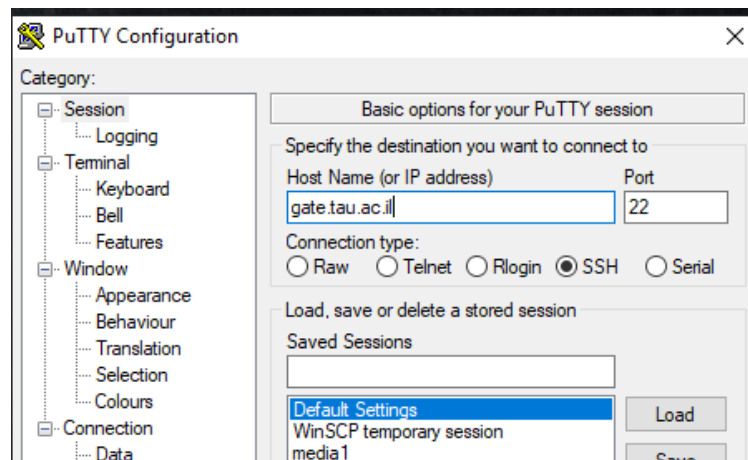

After connecting you can upload your files by dragging the designated file from your personal computer to the directory in the Nova server:
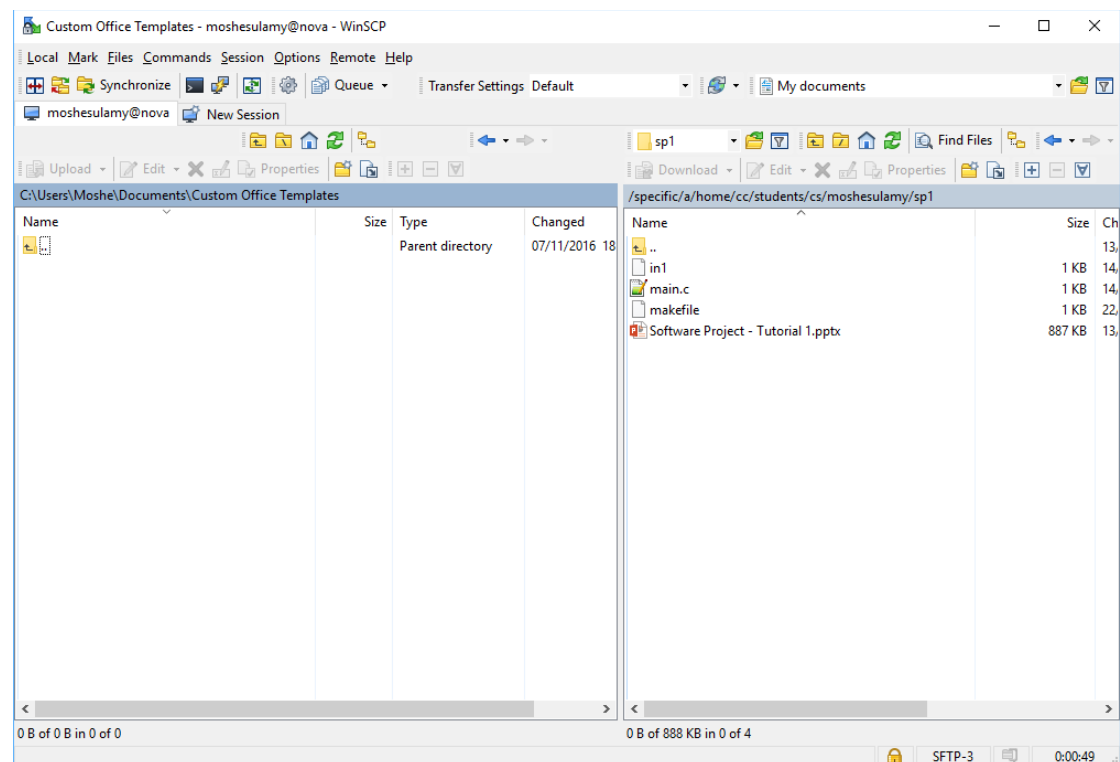
REMOTE CONNECTING TO NOVA

To connect to the Nova server, you can use PuTTY (*gate.tau.ac.il*):

After pressing open you will be asked to enter your username and password for authentication (the username and password are the same as in your moodle authentication). Once connected, you should enter the following command: ***ssh nova.cs.tau.ac.il***

When successfully connected to Nova you will be able to see the terminal as in the following picture:
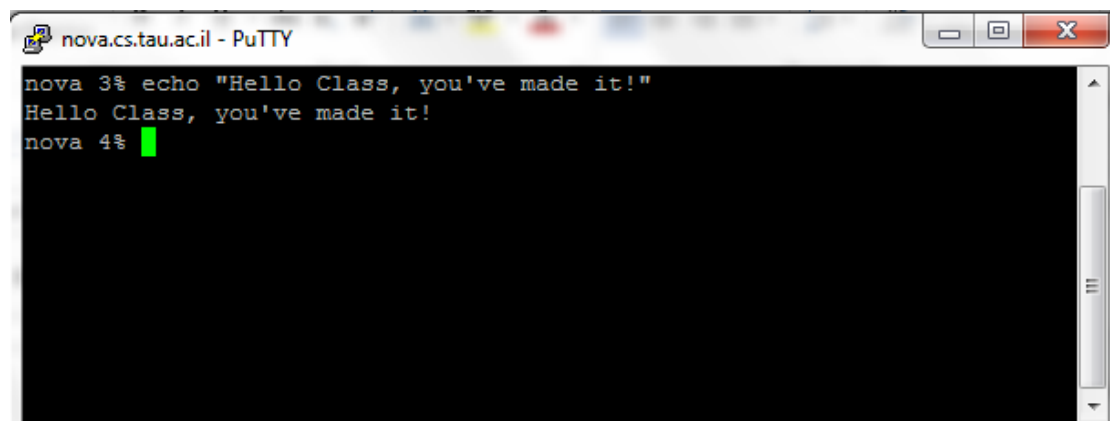
Now you can enter shell commands and execute them by pressing enter. Read the section below to find out more about shell commands.

(Note: echo is a shell command that is used to print strings to the standard output.)

USEFUL LINKS

You can download PuTTY and WinSCP by clicking on the [following link](#). Recall that we recommend that you work with eclipse; please follow the installation guidelines for eclipse in moodle.

BASIC SHELL COMMANDS

After the connection with the server (Nova) is established, you can now type in shell commands and execute them simply by pressing enter. The results will be shown on the terminal (If there's any).

A few useful shell commands:

>> pwd

Prints the full pathname of the current working directory to the standard output. (The pathname is relative to the root directory which is the first directory in Linux).

>> ls [dir]

Lists the content of the directory "*dir*" (Both files and directories). If no parameters are given, the result is the content of the current working directory.

>> cd [dir]

Changes the current directory to be *dir.*
Use the following shortcuts:

. – This is a shortcut for the current directory.
.. – This is a shortcut for the parent directory in the hierarchy.
~ – this is a shortcut for your home directory.

>> mkdir [dirName]

Creates a new directory with the name dirName.

>> cp [file1] [file2] … [fileK] [dir]

Copies *file1,file2,…,fileK* to the directory *"dir".*
Note: To copy an entire directory (recursively copy a directory) use –r flag.

>> rm [file1] [file2] … [fileK]

Removes *file1,file2,…,fileK.*

>>man [command]

The man command is used to display the manual page of the command "*command*".
Note: you can navigate through the manual page using the arrows in the keyboard. To exit the manual page press "*q*"

>> diff [file1] [file2]

Prints the difference between the two files (file1 and file2)
Note: Use the *man* command to see the manual page of the command *"diff".*

>> chmod [options] [file]

Changes the permissions of [file] according to [options].
Note: When transferring executable files to *nova*, you need to execute *"chmod 777 <file>"* to be able to execute it.

FIRST C PROGRAMS

In this assignment you will write your first C program. The program receives a number in base a and converts it to base b. For example, if you receive the number 101 in base **2** (binary base) and you are asked to convert it to base **10** (decimal) then the result should be 5.

MAIN

The main function is the entry point of any C program. We usually avoid implementing other functions inside the main.c file. However, in this assignment, you may implement auxiliary functions inside the source file **main.c**.

The behavior of the main function is as follows:

1- The program receives an integer representing base a (*hint*: simply use *scanf* with "*%d*"). The program asks the user to input the base by printing the following message:
   "Please enter the number's base:\n"
   a. If the user enters a base $a \notin [2,16]$ then the program prints the following error and terminates: **"Invalid input base\n"**
2- Afterwards, the program receives an integer representing base b (in the same manner). The program asks the user to input the base by printing the following message: **"Please enter the desired base:\n"**
   a. If the user enters a base $b \notin [2,16]$ then the program prints the following error and terminates: **"Invalid desired base\n"**
3- The program will ask the user to enter a number in base a by printing the following message:
   "Please enter a number in base <a>:\n"
   Where <a> is the value of the base a.
   For example if $a = 3$ then the message is: **"Please enter a number in base 3:\n"**
   a. The user will input the number as a series of chars. If any character is invalid (i.e., a character that is not in base a) then the program prints the following error and terminates: **"Invalid number!\n"**
   b. Use *getchar()* to read the user input for the number in base a.
4- Finally, the program converts the number from base a to base b, prints the following message: **"The result is: <res>\n"**, and terminate.
   Where <res> is the number received in **(section 3)** printed in base b.

Assumptions and requirements:

- The input number ends when a whitespace or EOF character is received. Ignore any input after the 3rd number (i.e., 4th input and beyond). For bases beyond 10, support both lowercase and uppercase letters. You may assume all *valid* input numbers can be represented with an *int* (in base 10), and no input exceeds 1024 characters.
- All input values are separated by any number and types of whitespace characters. You may not assume anything regarding the validity of any of the 3 input values provided by the user, they may even not be provided! Valid numbers are **integers only**, no floating numbers.
- If, at any point, there was an error while reading from the standard input then the program prints the following error and terminates: **"An error occurred!\n"**
- You may not change the supplied makefile or use functions from *math.h*
- Strange behavior may be observed due to using *scanf("%d",…)*, and that is expected and OK, as explained in Tutorial 1. For example, try executing with an input base of "7.3", "11x", etc.
- Any further questions should be answered by using the supplied executable. Use it!

As stated before, students may work on any development environment they choose. However, your code should compile and run on Nova. Please upload your source code (main.c) and the makefile provided in the assignment zip file to nova (note that all the files should be in the same directory).

After the connection is established, set your current directory as the source code's (main.c) directory. Type in the following command and press enter to build your program:
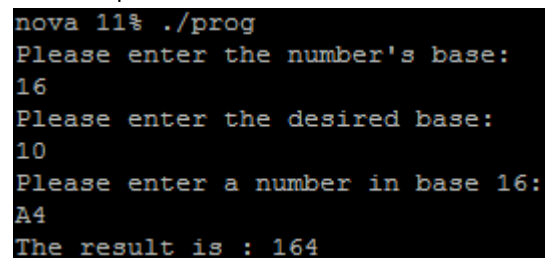
>> gcc -ansi -Wall -Wextra -Werror -pedantic-errors main.c -o ex1

The following command will compile your program and create a binary file called ex1. Note that the compilation flags used in the command above will be our default flags, so you should configure your eclipse installation with these flags (more info in moodle).

Now you can run your program by executing the following line on Nova:
>> ./ex1
See example below:

```
nova 11% ./prog
Please enter the number's base:
16
Please enter the desired base:
10
Please enter a number in base 16:
A4
The result is : 164
```

You can also check your program using the executable supplied along with the assignment by following these instructions:

1- Compile your program.
2- Use I/O redirection as follow:
   >> ./ex1 < test1.in > result1.out
   this will use "*test1.in*" as the standard input and write the results to the file "*result1.out*" (The file will be created if it doesn't exist.)
3- Create various input files for any test case you can think of and use the supplied executable file to create expected output files from these.
4- Compare your results with the expected result. The expected result for "*test1.in*" is "*expected1.out*".
   >> diff result1.out expected1.out
5- If **nothing** was printed, then your results match the results we got (good job). If not, try to debug and find the source of your mistake.
6- Repeat steps 1-5 for additional files you create with the supplied executable file.

Before submitting your code make sure your code compiles using the makefile provided in the zip file. To do so, follow these instructions:

1- Copy your source code along with the file "makefile" into the Nova server. Place the files in the same directory. (IF you have already done so in previous round, just ignore this)
2- Go the directory where you copied the files to, and simply write the following command:
   >> make

3- As an output, make sure the files "main.o" and "ex1" were added to the directory (use "ls" command). You can run your program by executing the file "ex1" as stated in previous sections.

Note: Please look at the makefile and read the guidelines in the course material for more information on makefiles. You will need to implement one on your own in future assignments.

SUBMISSION

Please submit a zip file named **id_assignment1.zip** replacing *id* with your actual **9-digit** ID. The zipped file must contain the following files:

- main.c – Your source code for the program.

REMARKS

- For any question regarding the assignment, please don't hesitate to post in the forum.
- Late submissions are not acceptable unless you have the lecturer approval.
- Borrowing from others' work is not acceptable and may bear severe consequences.

GOOD LUCK