Exercise 1

Submission due date: 15/11/2015 23:55

Introduction

The purpose of this assignment is to get familiar with Linux and developing 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 Program. The program would be a simple command line program which implements a basic calculator with the two subtraction and addition operations in addition to a special operation that will be defined later.
3- Compiling, Debugging and basic use of makefiles.

Developing Environment

Students may work in any developing environment they like, however we encourage you to use eclipse for developing and debugging. 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 will be described shortly. In addition your code will be manually checked and your grade will be affected if writing a "bad code".

File transferring

If you need to transfer files from your personal computer to nova, you can use WinSCP, to connect you will need to put your personal authentications. Please find the example bellow for your reference.
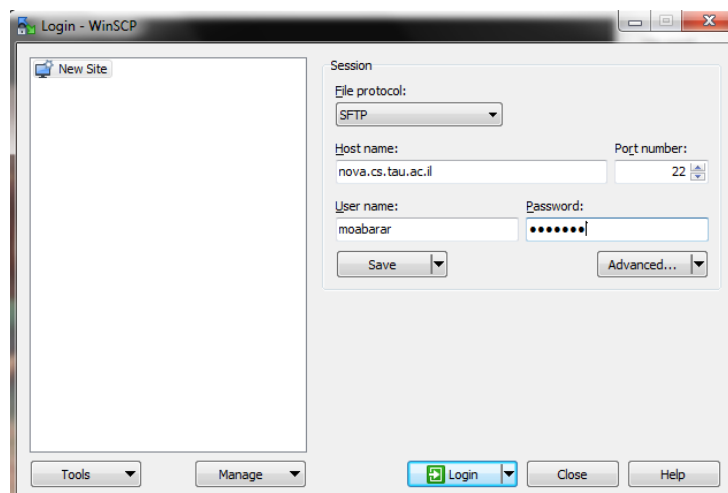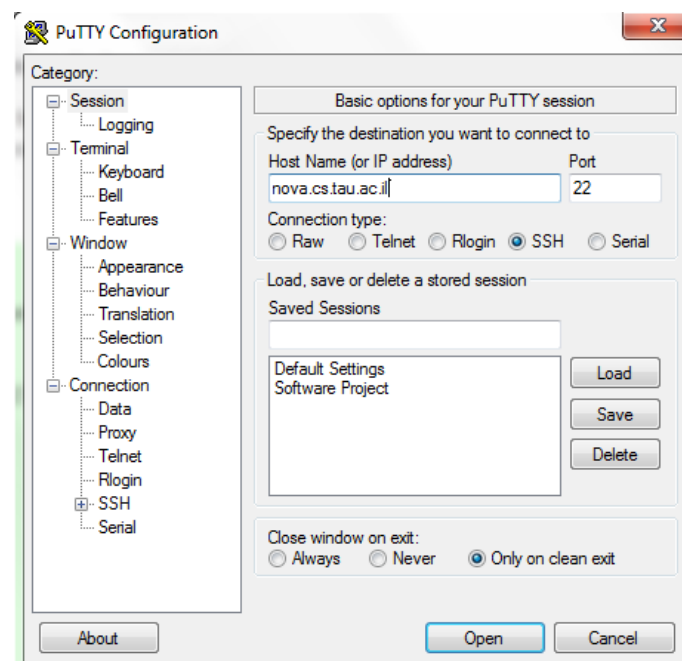

After connecting you can upload your files by dragging the designated file from your personal computer to the directory in nova server. Find the example bellow:
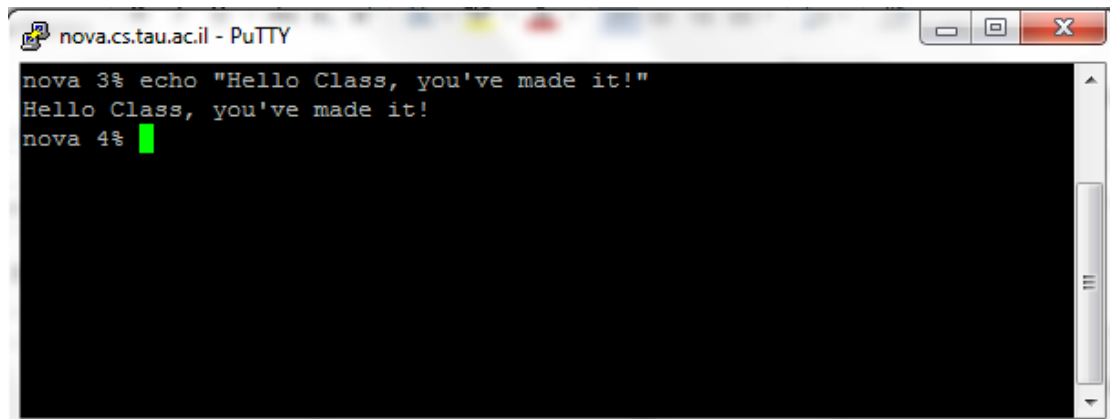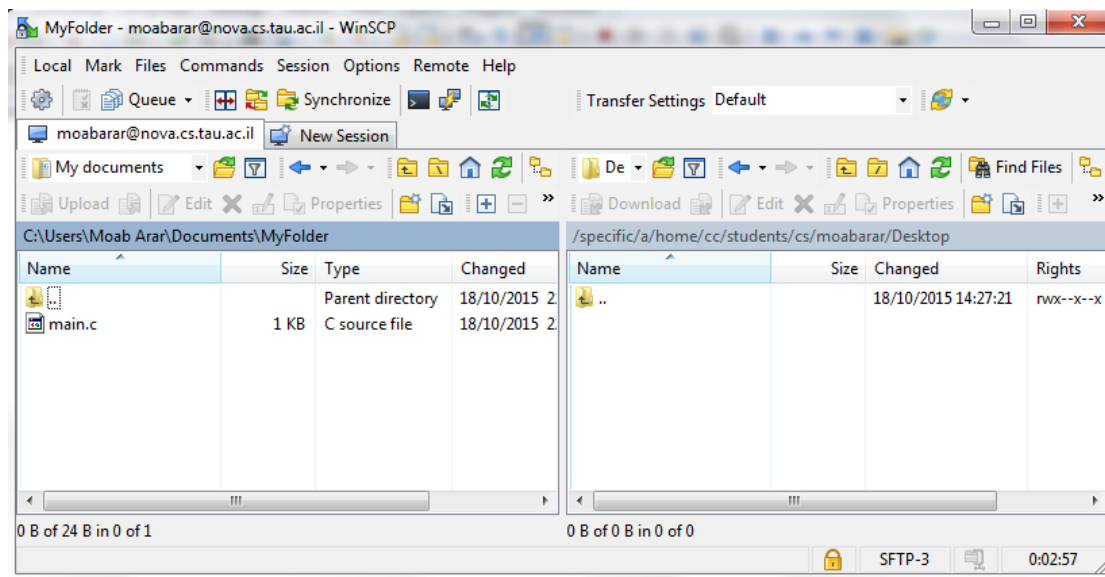
Remote Connecting to Nova

To connect to nova server you can use PuTTY to do so. Please see the bellow configuration for your reference:

After pressing open you will be asked to put your username and password for authentication. When successfully connecting to nova you will be able to see the terminal as in the bellow picture

Now you can write down shell commands and execute them by pressing enter. Please see the bellow section 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](#). Plus we recommend you download the following [eclipse version](#). Follow the installation guidelines for eclipse in the moodle website.

Basic shell commands

After connection is established with the server, you can now write shell commands and press enter to execute them. The results will be shown on the terminal (If there's any).

An example of shell commands are as follow:

>> 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 upper directory in the hierarchy.
"~" – this is a shortcut for the home directory.

>> mkdir [dirName]

The command create a new directory with the name dirName.

>> cp [file1] [file2] … [fileK] [dir]

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

> Examples:
>
> >> cp /dir1/myFile /dir2
>
> Copies "*myFile*" which is located in *"/dir1"* to the directory "*/dir2*"
>
> >> cp –r /sourceDir /destinationDir
>
> This copies the directory "*sourceDir*" (with its content) to the directory "*destinationDir*"

>> 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 *"sdiff".*

First C program

In this assignment you will write your first C program. You will be requested to implement a command line calculator that supports the two basic mathematical operations; addition and subtraction. Plus the program will support a new operation with the following sign '$'. The operation is defined as follow:

$$a \$ b = \begin{cases} not\ valid & a > b \\ a + (a + 1) + (a + 2) + \cdots + (a + (b - a) - 1) + b & a \le b \end{cases}$$

That is, the operation is defined such that a$b is the sum of all integers in the set [a,b]. Note that the operation is not defined when a>b. For Example:

1$3 = 1 + 2 + 3 = 6
1$1 = 1
2$1 = not valid

The program will work as follow:

1- The program will ask the user to choose the operation as follow:
   "Please choose an operation (+/-/$):"
2- The program will receive from the user the operation sign and will store it in a char type variable.
3- The program will check if the operation sign is valid (either '+' , '-' or '$'). If the operation sign is not valid the program will exit with the following message:
   "Invalid Operation".
   You can assume that the program will receive as an input at most one character.
4- The program will ask the user to enter the first number as follow:
   "Please enter the first operand:"
5- The program will receive from the user an integer and store the value in an int type variable. If the number is not valid (e.g. the user entered a character instead of a digit) the program will exit with the following message:
   "Invalid Number".
6- The program will ask the user to enter the second number as follow:
   "Please enter the second operand:"
7- The program will receive from the user an integer and store the value in an int type variable. If the number is not valid (e.g. the user entered a character instead of a digit) the program will exit with the following message:
   "Invalid Number".
8- The program will print as an output the result of the operation of the two numbers in the following format:
   "The Result is: res"
   where res is the result of the mathematical operation.
9- If the result for the new operation $ is not valid, the program will print the following:
   "The Result is: Not Valid"

The program should follow these requirements:

- **Each message should be followed by a new line, use the special character "\n".**
- Your program must consist of only one file named: main.c
- Your program must follow the printing format as in the instructions; you can use the expected output for your reference.

Compile, Debug and Makefile

As stated before, students may work on any developing environment they choose, however your code should compile and run on nova (The faculty server). So in order to compile the program you wrote, please connect to nova and copy all relevant files (main.c and makefile) to a directory of your choice (note that all the files should be in the same directory). In the terminal write the following command

`>> gcc -std=c99 -Wall -Werror -pedantic-errors  main.c -o ex1`

The following command will compile your program, and will create a binary file called ex1. You can run your program by writing the following line:
`>> ./ex1`

Another way to check your program is by using the input files that you were given in ex1.zip. To do so we will use I/O redirection, in our case we will run the program where the standard input is from a file (for instance test1.in) and the standard output will be another file (for instance result1.out). Use the expected output to check your results.

For example to check your homework you could follow these steps:

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- Compare your results with the expected result for "*test1.in*", the expected result for "*test1.in*" is "*expected1.out*".
   `>> diff result1.out expected1.out`
4- If no result was shown, then your results matches the results we got (Good job). If not, try to debug and find the source of your mistake.

You can also use the makefile provided to compile your code. Make sure your code compiles with the makefile you were given, in order to check so please follow these guidelines:

1- Copy your source code along with the file "makefile" into nova server. Place the files in the same directory.
2- Go the directory you copied the files to, and simply write the following command:
   `>> make`
3- As an output, make sure the two files "main.o" and "ex1" were added to the directory (use "ls" command). You can run you 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 makefile. You will be needed to implement one on your own in future assignments.

Submission

Please submit a zip file named **id1_id2_assignment1.zip** where id1 and id2 are the ids of the partners. The zipped file must contain the following files:

- main.c – Your source code for the program.
- partners.txt – This file must contain the full name, id and moodle username for both partners. Please follow the pattern in the assignment files. **(do not change the pattern)**
- makefile – the makefile provided in the assignment files.

Notes:

- **Both students** are asked to submit the assignment.
- You can back up your submission on Nova.

Remarks

- For any question regarding the assignment, please don't hesitate to contact Moab Arar by mail: moabarar@mail.tau.ac.il.
- Late submission is not acceptable unless you have the lecturer approval.
- Cheating is not acceptable.

Good Luck