Jordan University of Science & Technology
Department of Network Engineering and Security
NES416- Network Programming
Programming Assignment 2

Due Date: see course E-Learning site

Goal: Understand how to build a concurrent server and **deal with signals**

Description:

You need to re-implement assignment one such that the server serves clients concurrently. That is, the server (parent) creates a child for each client's connection request (you need to print the port and IP address of the client). The child keeps serving the client until when the client exits, at which time also the child exits appropriately.

The functionality starts by the **child server** sending the client a menu which contains function to choose from. The menu should look like the following

1) Add
2) Subtract
3) Divide
4) Multiply
5) Compute GPA
6) Exit

After receiving the menu, the client displays it on the screen waiting for user input (selection). For example, if the user selects 1 (first choice in the menu), it means add two integers. The user has to enter the two integers as done in HW#1, i.e. 3 + 5, and ignoring any number of extra spaces. The child server should perform the required operation and sends result back to the client, which is displayed there. Furthermore, the child server sends the menu again to be displayed at the client side for the user to enter his/her choice again, until 6 is selected at which time both the client and child server exits appropriately, where the parent prints the process ID of the terminated child.

For the GPA (Grade Point Average) calculation, you need to send the server a comma separated list containing the course grade (out of 100) and number of credit hours for that course. For example, if you send 80,3,90,1,65,3,70,2, -1 this means that you need to compute the GPA as the following, where -1 indicates the end of the list

Grade	Credit hours
80	3
90	1
65	3
70	2
-1	End of list
GPA= (80*3 + 90*1 + 65*3 + 70*2)/9 = 73.88	

Note that in all cases, the parent server is kept running.

The client program should use command line arguments to read the IP and Port numbers of the server. In addition, no need to use bind() at the client side. Furthermore, the server should use command line arguments to indicate the port it will listen to. Once the client prints the received reply from the server, it also prints the IP and port number of the server. Also, once the child server receives the clients requests, it is displayed together with the client's IP and port number

Sample output:

Similar to assignment one, except that when a child exits, it's PID is printed at the parent side

Hints:

- □ DO NOT use the header file "unp.h" from the book
- □ Each student should use a port number for the server such that is larger than 45000
- □ Your program for client needs to take two arguments that specify the IP address of the server and the port that it is trying to connect to. Your program for server needs to take an argument that specifies the port that it is listening to (the same one provided to the client)
- □ **Don't use bind() at the client side**
- □ Ask questions as early as possible.
- □ Submit your source code for both the client and the server, and some running sample of your code as one zipped file whose name is your student ID number
- □ Your programs should be compiled and run without any single error or warning.
- □ **Comment and error-check you code**