*This was translated to English using Google Translate, in order to upload to git.
The translation might not be accurate!

Exercise 1 in operating systems semester in 2023-4

Version 1.1 April 14, 2024

Themes work in Unix. dishes. Debug. Libraries, creating ,processespipes, file descriptor .

The purpose of the exercise

- Straightening the line in everything related to the use of Unix tools, building a library, checking the code anddebugging .Mike and the like ,
- Working with processes usingfork, exec
- Creatinga pipe usingpipe and copyingfd

General instructions

- The exercise can be submitted onlinux (in a virtual machine), ona mac) xcode andxcode command line tools must be installed or on (windows) WSL must be installed .(
  Exercises that will be submitted in a non-Linux environment - it is mandatory to use**the** POSIX**API** only.

It is forbidden to usethe API ofCOCOA (Apple) orWINDOWS .

- It is recommended to make sure that the code also runs inLinux in order to .save appeals in the test
- The exercise can be written inC orC ++
-) a separate librarycode and screenshots oradditional files . (
- Onetar.gz file must be submitted that contains all the sections of the exercise. You can read abouttar and (1)gzip on the Internet and in the (1) correspondingman pages. - See instructions on how to createa tar.gz at the .end
- All subtasks must be submitted
- Serving in pairs. Students who feel that they have been harmed by the situation (for example reservists, spouses of reservists, evacuees, etc.) may declare that they have been harmed by the situation for any reason and .submit in three
- All students in the group (individual, pair or trio) are responsible for the entire exercise they submitted including all sub-tasks and are required to defend .the entire exercise

Exercise 1 - Compilation and debugging in Linux - 10 points

Write 3 plans that fall in the following ways

1. Browsing from the stack (for example due to infinite recursion)
2. division by zero
3. Using undefined memory (reading or writing from an undefined address. For example 0xdeadbeef (

*This was translated to English using Google Translate, in order to upload to git.
The translation might not be accurate!

Createa core open the ,core usinga debugger model opening a ,core with and without debug info - ie flag)g in the compilation) open thecore using a textualdebugger - model where the drop is and value the variables using the where orprint command . Open thecore with a graphicaldebugger for example)ddd and model the crash with a graphical (debugger) If you don't have a graphical debugger installed, install it .sudo apt install ddd inubuntu (. .Submit - the code and screenshots of all steps

Exercise 2 - Using the library (learned with the help of the mathematical library) - 10 points

Write the*Poisson program* .that calculates the probability of a Poisson distribution
The program will receive 2 arguments (usingargc, argv which represent the values .(λ first) argument) andk .(second argument)
" If the program received more or less than 2 arguments the program will printError\n and " .exit
If the program received exactly 2 arguments it will print the valuepx(k that is (

$$p_X(k) = \Pr(X = k) = \frac{\lambda^k}{k!} e^{-\lambda}$$

with the precision ofa long double You are required to use) .expf ((3)
Please note that in order to use the functions of the mathematical library, you must compile with the-lm flag in addition, you must add to the calling code the ,header of the library - in theC language the name of theheader ismath.h in the languagec the name of the ++ header iscmath .
It is required to submitmake and code and a screenshot of a running example

Exercise 3 - Building a library - 10 points

Using exercise 2, construct thepoisson function⁻ .that calculates the Poisson distribution
Compile it into a dynamic library (ieshared object called (libpoisson.so
Write a program that uses the library and calculates apoisson distribution for 5 values

λ	k	value
2	1	1
2	10	2
2	2	3
3	3	4
100	3	5

and prints them in 5 lines (each value in a separate line)
It is required to submitmake and code and a screenshot of a running example

- Exercise 4code coverage weight 20 points -

*This was translated to English using Google Translate, in order to upload to git.
The translation might not be accurate!

At https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/
You will find a (working) implementation of Dijkstra's algorithm. (You can chooseC orC (++
(;;) Change the program so that your program supports (within the for loopreceiving a new
graph, (reading the graph will be done fromstdin usingscanf orcin (of your choice
correctness check (i.e. I didn't put too many or too few distances in the line. Dijkstra does
.not support arc weights negatives) and running the algorithm
Check the program, check your test with (1)gcov .
Show that your test covered all the code you submit including the edge cases
(.ie incorrect input - too many arcs in a line, too few lines. arcs with negative weight, etc)
,You must submit - codemake , gcov .output (1) a screenshot of the run

- Exercise 5**profiling** weight 20 points -

Realize the three solutions to themax sub array sum problem. (n, n ^{2}, n ^{3})
:Your programs will receive two arguments
- Onerandom seed for use with)srand (
Second - the size of the input (the amount of numbers the program will generate)
The input to the three algorithms will be generated randomly (using calls to (3)rand and (
the running of the algorithm will also be done in the function. The random numbers can be
.uniformly distributed in the segment (-25, 74)
If you want a uniform distribution otherwise note that negative numbers must be included)
(otherwise the complete subsection will be the complete subsection
Run the three solutions on input size 100, 1000, 10000
Demonstrate the running time of the algorithm versus the time of generating the random
numbers usinggprof .(1)
.It is required to submit the code of the three programsMake ,screenshot ,gprof outputs .
Note: The problem definition and the three algorithms can be found on pages 3-21 of
https://cses.fi/book/book.pdf

Exercise 6 - usingpipes .creating processes - 30 points ;

The purpose of this exercise - work withpipe(2), execve(2), fork and not work with (2)
strings.
An exercise that implements the problem usingstrings api .will be disqualified
.I implement a phone book using a text file
:In a text file I have all the names together with the phone number in the following structure
a line for each name and number, separated by a comma between the name and the)
(number, the line ends with a new line
Nezer Zaidenberg, 054-5531415\n
.The text file can contain dozens and hundreds of records in this format
- For this purpose I actually the following 2 programs
add2PB .which adds a new entry to the phonebook (just a new line) -
The program will usually accept a name - first and last name but which can contain spaces
for example in the case of)Bat sheva or a middle name). Can only contain a first name (if
the last name is unknown to us or for names likemom, dad then a comma (we are (
guaranteed that a name never contains a comma) then the phone number. The end of a
record will always bea line feed .
findPhone which finds the phone of the person received in -argv by calling commands (1)
cat(1), grep(1), awk(1), sed(1), cut(1)
Create processes (usingfork and (2)execXX and copy (2)file descriptors usingdup or (2)
dup (2)2
submitmake and code for both programs as well as a running example (for example a
(screenshot

*This was translated to English using Google Translate, in order to upload to git.
The translation might not be accurate!

Note - you can read aboutgrep, awk, sed . at the following address
https://www-users.york.ac.uk/~mijp1/teaching/2nd_year_Comp_Lab/guides/grep_awk_sed.pdf
For the avoidance of doubt - this exercise should be written inC orC and not in ++bash or PERL
.An important relief - it can be assumed that each person has only one phone number
In addition, it can be assumed that I only know one person in each name. If by chance I
.asked for a name that appears in the phone book twice the answer could be any answer
For example if I know two people namedAvner and the file contains
Avner Harishon, 03-1234567
Avner Hasheni, 050-9876543
Any answer is possible (including no answer). In addition, it can be assumed that the
.character # does not appear as part of the name or number on any phone
A possible solution to the problem of finding a phone number
grep “Mickey Mouse” phonebook.txt | sed 's/ /#/g' | sed 's/,/ /' | awk {print$2}
.The first command will return only the line that contains Mickey Mouse
.The second command will turn all profits into scales
The third command will create a space instead of a comma (thus creating a second column)
The fourth command will print the second column (ie the phone)

Appendix working with**tar** and**gzip**

- In 2 commandsOLD SCHOOL

(1) tar ortape archive Collected several files (in a folder for example) and pasted them to .
each other using the command
tar -cvf mytarfile.tar mydirectory
To open the file we used the
tar -xvf mytarfile.tar
We often wanted to compress (zip) the files one of the popular compression applications
was (1)gzip or)gnuzip (
To compress we used b
gzip myfile
.Which would create a compressed file and add an extensiongz
To open we used the
gunzip myfile.gz
- Notegnuzip is a popular and fast compressor but there are other compressors (some compress better) for example
bzip2(1), xzip(1), compress(1)

in one command

Modern versions of (1)tar also know how to compress
tar -zcvf mycompressedfile.tgz mydirectory
will compress the contents ofmydirectory intomycompressedfile.tgz
To open we will use the command
tar -zxvf mycompressedfile.tgz
There are other flags for replacing the compressor for details seeman tar

It is important

*This was translated to English using Google Translate, in order to upload to git.
The translation might not be accurate!

The weight of the exercise is 10% of the final
 . grade in the course
The weight of the protection - 5% more. Treat the
!exercise accordingly