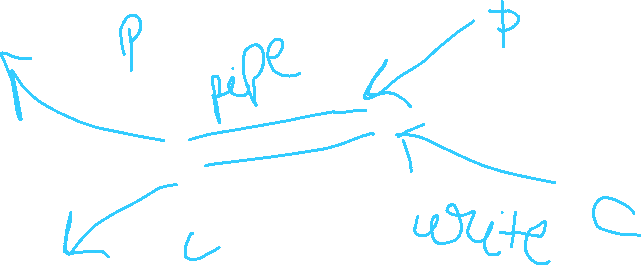
LAB 8

* Working with fifos this lab
* Differences between pipes and fifos
* Fifos = extended version of pipes
* Summary of the last lab:
* Pipes->parent and child own ->reading and writing end
* Evry time when we use a specific operation we have to close each end

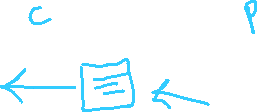


->pipe ->think about fork processes

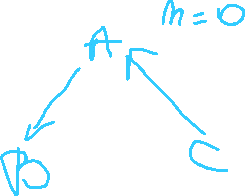
->c program between processes using pipes=>use fork()

->FIFO = named pipe

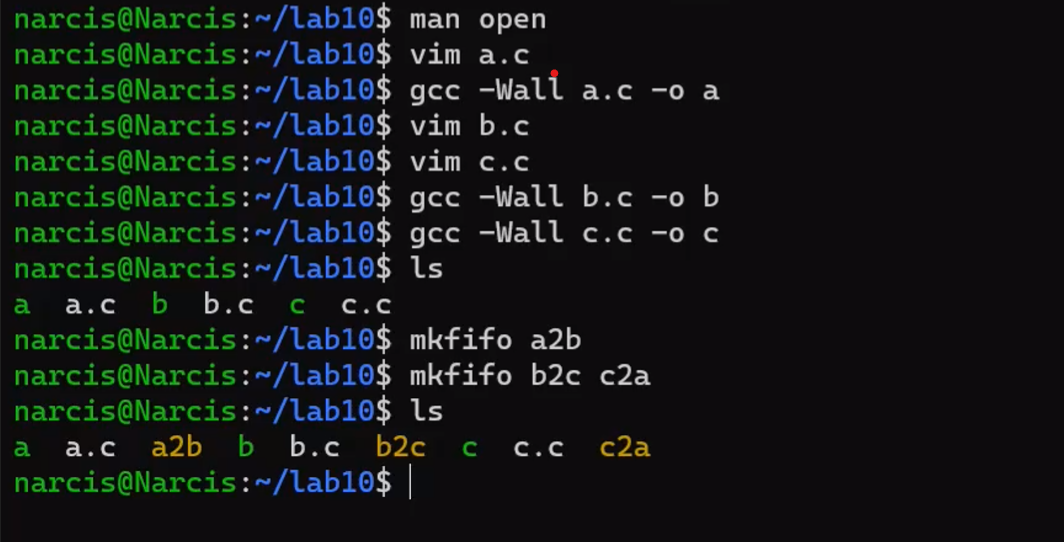
->the pipe is not stored in the memory, but it is a SPECIAL file that we are creating on the disc->we have to open, read, write ->are not open by default



* Once file is opened to write it will stay open until we give another open mode
* When we do not need a fifo, we close it
* EXAMPLE FROM LAST LAB: but using fifos



* We use 3 different c programs
* Use one terminal
* Start by coding program a.c
* Include libraries
* Main function that we will have to declare a var n=20(initialized with 20), we need 3 fifos
* A2b = open(“a2b”,O\_WRONLY) -> reading
* c2a = open(“c2a”,O\_RDONLY) ->write
* Keep reading writing ->while that will go to infinity(because we do not know which process is going to end first)
* Read->stay unblocked until there are no more writers active ->read from fifo
* If read <=0 =>fifo closed =>break the code
* N==0 =>process closed =>break the code
* Close fifos
* Copy everything in c.c, b.c from a.c because we will only change the fifo names
* Mkfifo a2b b2c c2a -> creates a fifo
* Add <fcntl.h> library in all c files
* Compile with: gcc -Wall a.c -o a



A screen shot of a computer program

Description automatically generated with medium confidence

A screen shot of a computer program

Description automatically generated with medium confidence

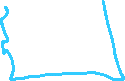
* Open 2 more terminals
* Terminal 1 => executes ./a =>does nothing =>because a tries to open the fifos
* ./b
* ./c =>everything stars to flow

A screenshot of a computer

Description automatically generated

* Check the open <0 ->print a message that it is an error !!close the fifos!!!!
* TEST -> maximum of 3 processes
* 2 types of problems with pipes and fifos
* A b c d schema
* Fixed nr of processes -> solve the problem with pipes(fork!!),fifos(separate c programs) explicitly pipes/fifos
* A process -> parent/process a read values and send them to the other processes-> do something with them ->at the end send them back to the parent

->example:



* Pipes ->first close the pipes that we do not need!!!
* 0->read 1->writing
* Write ->immediately after we need to use a read!!

#include <stdio.h>  
#include <stdlib.h>  
#include <unistd.h>  
#include <sys/wait.h>

int main()  
{  
        int a, b, s, p;  
        int p2c[2], c2p[2];

        pipe(p2c); pipe(c2p);

        int pid = fork();

        if (pid == 0)  
        {  
                close(p2c[1]); close(c2p[0]);

                while (1)  
                {  
                        if (read(p2c[0], &a, sizeof(int)) <= 0)  
                                break;

                        if (read(p2c[0], &b, sizeof(int)) <= 0)  
                                break;

                        s = a + b;  
                        p = a \* b;

                        write(c2p[1], &s, sizeof(int));  
                        write(c2p[1], &p, sizeof(int));  
                }

                close(p2c[0]); close(c2p[1]);

                exit(0);  
        }

        close(p2c[0]); close(c2p[1]);

        while (1)  
        {  
                scanf("%d %d", &a, &b);

                write(p2c[1], &a, sizeof(int));  
                write(p2c[1], &b, sizeof(int));

                read(c2p[0], &s, sizeof(int));  
                read(c2p[0], &p, sizeof(int));

                printf("Sum: %d\nProduct: %d\n\n", s, p);

                if (s == p)  
                        break;  
        }

        close(p2c[1]); close(c2p[0]);

        wait(0);  
        return 0;  
}

A picture containing text, screenshot, font

Description automatically generated->runs

* 2 processes but using fifo: parent=a, child=b

A picture containing text, screenshot, display, software

Description automatically generated

* Recomandations for test prep on teams 913 channel ->take all examples form the lab
* 30 more examples, but not the ones using signals ->all those problems have the solutions 😊
* Pipe, fifo, fork processes
* Max 3 processes ->communicate between them using fifo or pipes
* Reading form keyboard
* Stopping conditions