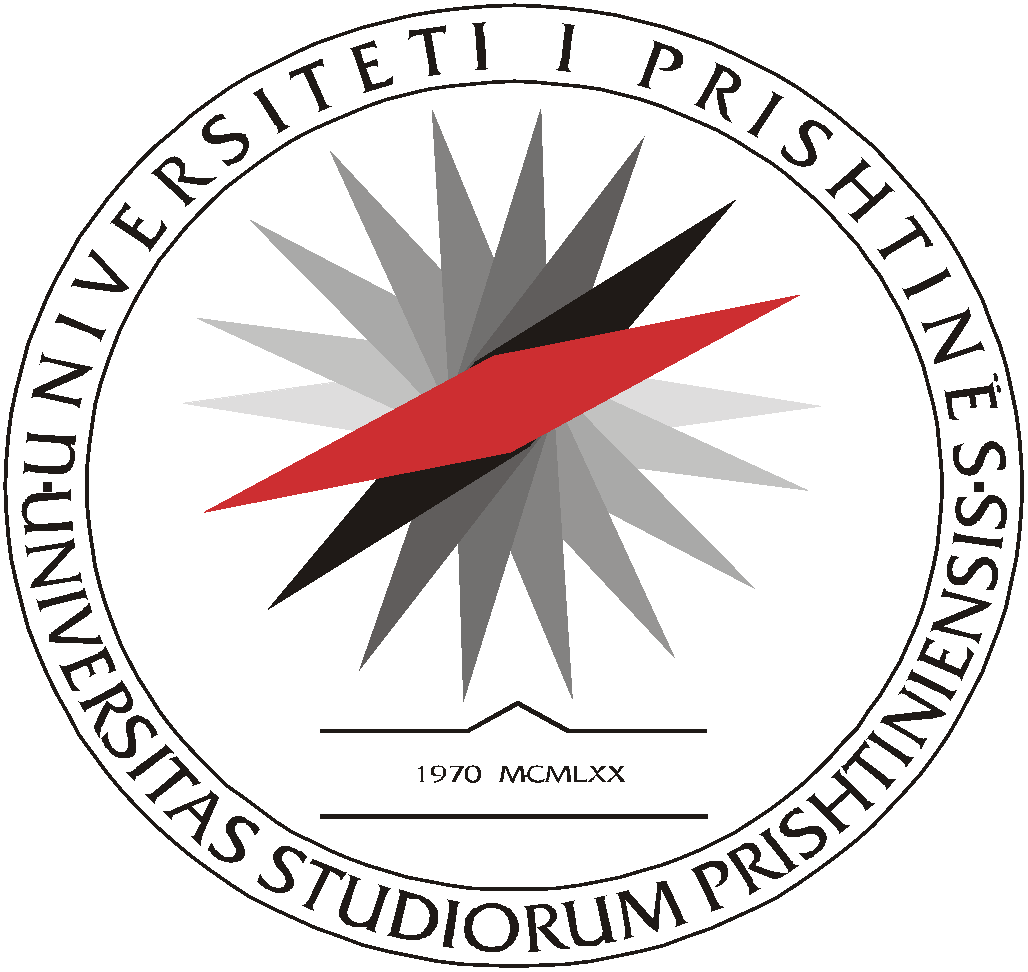
**Universiteti i Prishtinës**

Fakulteti i Inxhinierisë Elektrike dhe Kompjuterike



**Seminar nga lënda: Sisteme Operative**

**Komunikim mes proceseve (IPC) përmes anonymous pipes**

Profesor: Punuan:

Prof.Ass Artan Mazrekaj Çlirim Hodaj,

Aurel Demiraj,

Betim Thaçi, Besar Pallush, Drinos Kukaj.

Prishtinë, 2021.

Përmbajtja

[Hyrje 3](#_Toc61806216)

[Pipes 3](#_Toc61806217)

[Përshkrimi i problemit 4](#_Toc61806218)

[Teknologjitë e përdorura 5](#_Toc61806219)

[Implementimi 6](#_Toc61806220)

[Kodi ne fajllin parentp.c 6](#_Toc61806221)

[Kodi në fajllin child.c 8](#_Toc61806222)

[Rezultati 9](#_Toc61806223)

[Përfundimi 10](#_Toc61806224)

[Literatura 11](#_Toc61806225)

# Hyrje

Në momentin kur një sistem,aplikacion apo një proces dëshiron të komunikojë me një sistem, aplikacion apo ndonjë proces tjetër, shfrytëzon IPC (Inter-process communication).Inter-process communication (kouminikimi ndërprocesor) është një mekanizëm që lejon komunikimin ndërmjet proceseve. Mekanizmat që shërbejnë për komunikim ndërmjet proceseve në Windows janë:

* **Pipes.**
* **File Mapping.**
* **Mailslots.**
* **Windows Sockets.**
* **Clipboard.**

## ****Pipes****

**Në këtë detyrë do të testohet shembulli i komunikimit ndërmjet proceseve me anë të pipes-përkatësisht Anonymous Pipe. Kemi dy lloje të "pipes":**

1. Ordinary Pipes (**Anonymous Pipe në Windows**)
2. Named Pipes

**Anonymous Pipe janë pipe të paemërtuara të cilat ofrojnë komunikim një-kahor. Ky lloj i pipes është një drejtimësh. Në parim, komunikimi kryhet ndërmjet procesit prind dhe atij fëmijë (i krijuar nga prindi).**

**Nëse kërkohet komunikim i dyanshëm, duhet të përdoren dy tuba (pipe), ku secili tub dërgon të dhëna në një drejtim tjetër.**

Nëse lejohet komunikimi dykahësh, është **half-duplex** (të dhënat mund të transmetohen vetëm në një drejtim në të njëjtën kohë) apo **full-duplex** (të dhënat mund të transmetohen në dy drejtime në të njëjtën kohë).

Named Pipes janë metodë më e avancuar për komunikim ndërmjet proceseve. Lejojnë kominikim edhe me anë të rrjetit. Mundësojne komunikimin dy drejtimësh, full-duplex etj.

Por në këtë rast do të implementojmë **Anonymous Pipe.**

* **Ky lloj i pipes lejon komunikim në stilin standard të prodhuesit-konsumatorit.**
* **Prodhuesi shkruan në një skaj (write-end i pipe)**
* **Konsumatori lexon nga skaji tjetër (read-end i pipe)**
* **Tubat e zakonshëm janë njëdrejtimësh**
* **Kërkojnë relacionin prind-fëmijë në proceset komunikuese**

# Përshkrimi i problemit

Problemi që duhet të zgjidhet është realizimi i komunikimit dy drejtimësh përmes pipes në mes të procesit prind dhe atij fëmijë. Me anë të këtij programi, të dy proceset do të mund të dërgonin të dhëna në njëri-tjetrin brenda së njëjtës kohë. Pra, me ekzekutimin e procesit prind dërgohet mesazhi i tij ne procesin fëmijë e në të njëjtën kohë merret edhe mesazhi që ka dërguar procesi fëmijë. Në ekran do të shfaqen mesazhet e dy proceseve ndaj njëri-tjetrit.

Pasi që anonymous pipes nuk përkrahin komunikimin dy drejtimësh, duhet që të krijojmë dy tuba (pipe). Për krijimin e një pipe nevojitet funksioni **CreatePipe()** me katër parametra. Këta parametra përfshijnë handle-erët për lexim dhe shkrim, një instancë të strukturës **STARTUPINFO** e cila specifikon vetitë e sistemit si dhe madhësinë e tubit(pipe) në bajtë(parametër i katërt). Leximi dhe shkrimi në pipe mund të realizohet me funksionet e zakonshme **ReadFile()** dhe **WriteFile().**

Për dallim nga sistemi operativ Linux ku pipe trashëgohet në mënyrë automatike nga procesi fëmijë, në Windows duhet që të specifikojmë cilat atribute dëshirojmë ti trashëgojë procesi fëmijë.

Në këtë detyrë kemi krijuar dy programe që paraqesin procesin prind dhe atë fëmijë. Fillimisht kemi krijuar dy pipe dhe katër handles në procesin prind dhe më pas i kemi trashëguar vetëm dy handles në procesin fëmijë. Në procesin prind është deklaruar dhe inicializuar mesazhi "Pershendetje" që transmetohet në procesin fëmijë. Gjithashtu edhe në procesin fëmijë kemi deklaruar mesazh që transmetohet në drejtimin e kundërt (nga procesi fëmijë në atë prind).

Funksioni **CreatePipe** përdor strukturën **SECURITY\_ATTRIBUTES** për të krijuar handle të trashëgueshme në skajet e leximit dhe shkrimit të dy tubave(pipes). Skaji i leximit i një tubi shërben si hyrje standarde (standard input) për procesin fëmijë, ndërsa skaji i shkrimit të tubit tjetër është dalje standarde (output standard) për procesin fëmijë. Handles specifikohen me anë të strukturës **STARTUPINFO**, e cila i bën ata të trashëgueshëm nga procesi fëmijë. Kjo strukturë gjithashtu edhe ndalon trashëgimin e handles në procesin fëmijë.

Procesi prind përdor skajet e kundërta të këtyre dy tubave për t'i shkruar procesit fëmijë dhe për të lexuar mesazhin që dërgon procesi fëmijë. Siç specifikohet në strukturën **STARTUPINFO**, edhe këto handle të procesit prind janë gjithashtu të trashëgueshme. Sidoqoftë, këto handles nuk guxojnë të trashëgohen. Prandaj, përpara se të krijojmë procesin fëmijë, procesi prind përdor funksionin **SetHandleInformation** për të siguruar që handle i shkrimit (Write Handle i procesit prind) për hyrjen standarde të procesit të fëmijës dhe handle i leximit (Read Handle i procesit prind) të mesazhit të procesit fëmijë nuk mund të trashëgohen.

Siç mund ta shohim edhe në figurë vetëm handles që ndodhen në anën e procesit fëmijë trashëgohen në atë proces, ndërsa ata në anën e kundërt shfrytëzohen vetëm nga procesi prind.

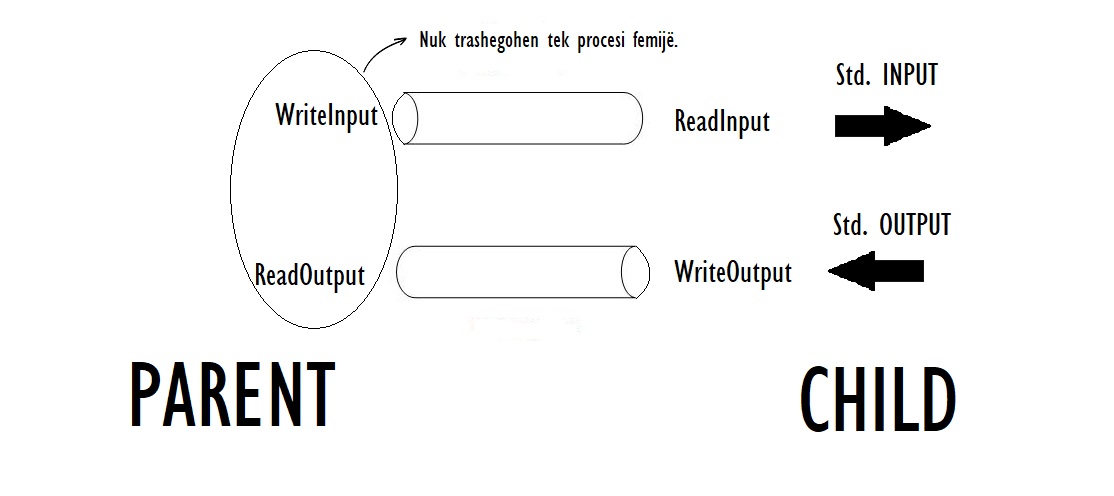
****

Fig . Transmetimi i të dhënave me dy tuba (pipes).

# Teknologjitë e përdorura

Për realizim të detyrës është përdorur:

* Editori Visual Studio Code,
* Gjuha programuese C,
* Sistemi operativ Windows 8.1.

# Implementimi

### Kodi ne fajllin parentp.c

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

#define BUFFER\_SIZE 25

int main(VOID)

{

   HANDLE ReadHandle, WriteHandle; // per hyrje teksti ne child

   HANDLE ReadHandleO, WriteHandleO; // per dalje teksti nga child

   STARTUPINFO si; // krijimi i instances se STARTUPINFO (specifikon vetite e sistemit)

   PROCESS\_INFORMATION pi; // strukture qe identifikon procesin e ri te krijuar

   char message[BUFFER\_SIZE] = "Pershendetje"; // mesazhi qe do te dergohet ne procesin femije

   DWORD written; // pointer per shkrim ne pipe

   /\* percaktimi i atributeve te sigurise qe i mundeson pipe(tubave) te trashegohen \*/

   SECURITY\_ATTRIBUTES sa = {sizeof(SECURITY\_ATTRIBUTES), NULL, TRUE};

   /\* alokimi i memories \*/

   ZeroMemory(&pi, sizeof(pi));

   /\* krijojme njerin pipe (pipe1) \*/

   if (!CreatePipe(&ReadHandle, &WriteHandle, &sa, 0))

   {

      fprintf(stderr, "Krijimi i pipet te input-it per child deshtoi");

      return 1;

   }

   /\* Sigurohemi qe te mos trashegojme tek procesi femije writeHandle te pipe1 \*/

   SetHandleInformation(WriteHandle, HANDLE\_FLAG\_INHERIT, 0);

   /\* krijojme pipe-in tjeter per lexim nga child procesi (pipe2) \*/

   if (!CreatePipe(&ReadHandleO, &WriteHandleO, &sa, 0))

   {

      fprintf(stderr, "Krijimi i pipet te output-it per child deshtoi");

      return 1;

   }

   /\* Sigurohemi qe te mos trashegojme tek procesi femije readHandle te pipe2 \*/

   SetHandleInformation(ReadHandleO, HANDLE\_FLAG\_INHERIT, 0);

   // Percaktojme anetaret e struktures STARTUPINFO (specifikon vetite e procesit te krijuar)

   ZeroMemory(&si, sizeof(STARTUPINFO));

   si.cb = sizeof(STARTUPINFO);

   si.hStdError = WriteHandleO;

   // ridrejton daljen standarde per shkrim nga skaji i pipe1

   si.hStdOutput = WriteHandleO;

   // ridrejton hyrjen standarde per lexim nga skaji i pipe2

   si.hStdInput = ReadHandle;

   si.dwFlags = STARTF\_USESTDHANDLES;

   /\* krijimi i procesit femije \*/

   if (!CreateProcess(NULL, "childp.exe", NULL, NULL, TRUE, /\* trashegon handles \*/ 0, NULL, NULL, &si, &pi))

   {

      printf("Deshton krijimi i procesit\n");

   }

   /\* mbyllen skajet e paperdorura te pipes \*/

   CloseHandle(ReadHandle);

   CloseHandle(WriteHandleO);

   /\* procesi prind shkruan ne pipe \*/

   if (!WriteFile(WriteHandle, message, BUFFER\_SIZE, &written, NULL))

      fprintf(stderr, "Mesazhi nuk u shkrua ne pipe.");

   /\* mbyllet writeEnd i pipe1 \*/

   CloseHandle(WriteHandle);

   /\* procesi prind lexon mesazh nga procesi femije dhe ndalon kur nuk ka me mesazhe \*/

   DWORD dwRead;

   CHAR chBuf[BUFFER\_SIZE];

   BOOL bSuccess = FALSE;

   HANDLE hParentStdOut = GetStdHandle(STD\_OUTPUT\_HANDLE);

   for (;;)

   {

      // Lexon mesazhin e procesit femije nga pipe2

      bSuccess = ReadFile(ReadHandleO, chBuf, BUFFER\_SIZE, &dwRead, NULL);

      if (!bSuccess || dwRead == 0) break;

      // shkruan ne handle te tij per te shfaqur tekstin ne ekran

      bSuccess = WriteFile(hParentStdOut, chBuf,

         dwRead, &written, NULL);

      if (!bSuccess) break;

   }

   /\* Procesi prind pret derisa te perfundoje se ekzekutuari procesi femije \*/

   WaitForSingleObject(pi.hProcess, INFINITE);

   CloseHandle(pi.hProcess);

   CloseHandle(pi.hThread);

   return 0;

}

### Kodi në fajllin child.c

#include <stdio.h>

#include <stdio.h>

#include <windows.h>

#define BUFFER\_SIZE 25

int main(VOID)

{

HANDLE ReadHandle; // handle per lexim ne procesin femije

HANDLE WriteHandleO; // handle per shkrim ne procesin femije

CHAR buffer[BUFFER\_SIZE]; // ruhet mesazhi qe dergon procesi prind

DWORD read; // pointer per lexim te mesazhit

DWORD written; // pointer per shkrim te mesazhit

/\* Procesi merr readHandle-in e trasheguar nga pipe1 (inicializon ate me standardin e hyrjes) \*/

ReadHandle = GetStdHandle(STD\_INPUT\_HANDLE);

/\* Procesi merr writeHandle-in e trasheguar nga pipe2 (inicializon ate me standardin e daljes) \*/

WriteHandleO = GetStdHandle(STD\_OUTPUT\_HANDLE);

// Dergojme tekst ne daljen standarde te procesit edhe permes "printf"

    printf("\n \*\* mesazhi\_i\_child\_per\_parent \*\* \n");

/\* Procesi femije lexon mesazhin e procesit prind nga pipe1 \*/

if (ReadFile(ReadHandle, buffer, BUFFER\_SIZE, &read, NULL))

printf("Procesi femije lexoi: %s",buffer);

else

fprintf(stderr, "Procesi femije deshtoi te lexoje nga pipe");

// procesi femije dergon mesazhin per procesin prind permes pipe2

WriteFile(WriteHandleO,"Hello from child", BUFFER\_SIZE, &written, NULL);

return 0;

}

## Rezultati

Pasi të kemi përfunduar programin e procesit prind dhe fëmijë ekzekutojmë fillimisht procesin prind me anë të komandës në njërin prej terminaleve GitBash ose në terminal të VSCode

gcc –o parentp parentp.c

./parentp

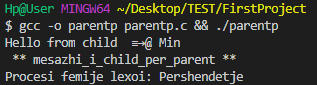


Fig . Rezultati pas ekzekutimit të fajllit parentp.c në terminal të VSCode.

Ekzekutimi i procesit prind në CMD bëhet përmes komandave të mëposhtme, meqë rast shfaqet rezultati si në figurën 3.

gcc –o parentp parentp.c

parentp.exe

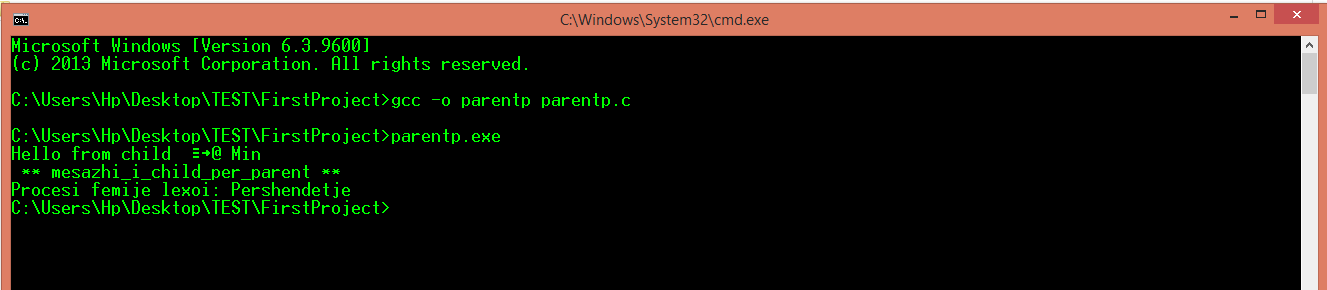


Fig . Rezulati pas ekzekutimit të fajllit parentp.c me anë të CMD.

# Përfundimi

Në këtë detyrë kemi hasur në literaturë jo të mjaftueshme për këtë lloj komunikimi në gjuhën C në sistemin operativ Windows. Por pas përkjekjeve e hulumtimeve kemi arritur të krijojmë komunikim ndërmjet proceseve përmes tubave të thjeshtë (anonymous pipe). Implementim më i lehtë i këtij lloj komunikimi bëhet në sistemin operativ Linux. Zgjidhja optimale e këtij problemi është implemementimi i tubave te emërtuar (named pipe), por sipas kërkesës së detyrës kemi arritur të krijojmë komunikimin dy-kahor edhe përmes anonymous pipe.

Gjithashtu, me rastin e përdorimit të gjuhës C jemi njoftuar me sintaksën bazë të saj dhe me metodat built-in të cilat i ka kjo gjuhë programuese.

Top of Form

Bottom of Form

# Literatura

1. Operating Systems Concepts 10 th edition, Abraham Silberschatz
2. https://docs.microsoft.com/en-us/windows/win32/procthread/creating-a-child-process-with-redirected-input-and-output
3. https://docs.microsoft.com/en-us/windows/win32/ipc/anonymous-pipe-operations
4. https://docs.microsoft.com/en-us/windows/win32/ipc/anonymous-pipe-security-and-access-rights
5. https://www.codeproject.com/Questions/5278701/A-TEXT-macro-error-in-a-console-app-of-visual-Cplu
6. https://www.codeproject.com/Articles/4816/Unleashing-anonymous-pipes-Part-1#:~:text=An%20anonymous%20pipe%20provides%20a,write%20end%20to%20the%20pipes.