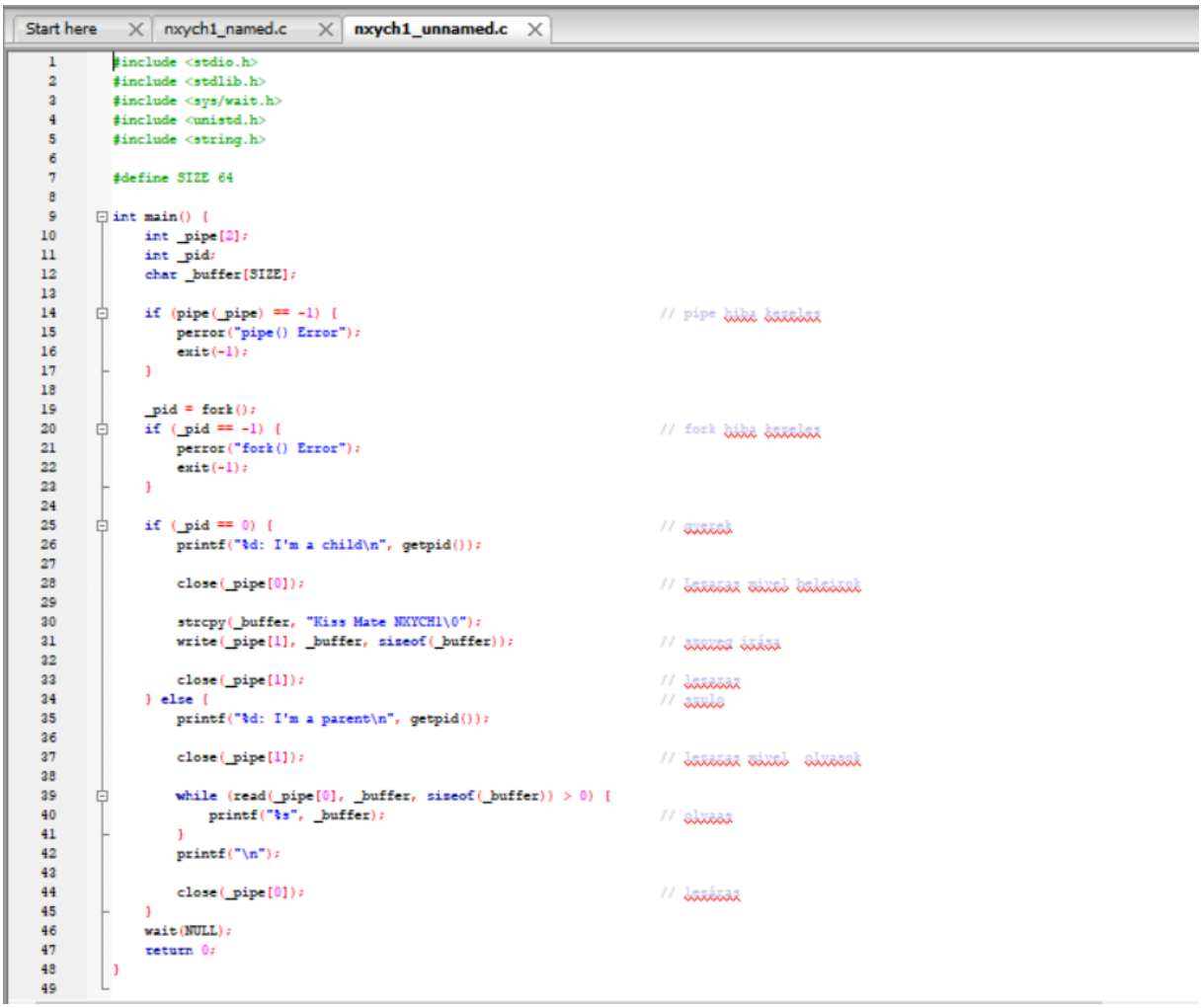
1. Feladat :
   . Készítsen C nyelvű programot, ahol egy szülő processz létrehoz egy csővezetéket, a gyerek processz beleír egy szöveget a csővezetékbe (A kiírt szöveg: XY neptunkod), a szülő processz ezt kiolvassa, és kiírja a standard kimenetre. Mentés: neptunkod_unnamed.c



2. Feladat: .

Készítsen C nyelvű programot, ahol egy szülő processz létrehoz egy nevesített csővezetéket (neve: neptunkod), a gyerek processz beleír egy szöveget a csővezetékbe (A hallgató neve:pl. Keserű Ottó), a szülő processz ezt kiolvassa, és kiírja a standard kimenetre. Mentés: neptunkod_named.c

Start here | nxych1_named.c | nxych1_unnamed.c

```c
#include <stdio.h>
#include <sys/file.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <signal.h>
#include <sys/wait.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

#define NAME "NXYCH1"
#define SIZE 64

int main() {
    pid_t _pid;
    int _fifo;
    int _open;
    char _buffer[SIZE];
    int _temp;

    _pid = fork();
    if (_pid == -1) {
        perror("fork() Error");
        exit(-1);
    }

    if (_pid == 0) { // ez a child
        printf("%d: I'm a child\n", getpid());

        _open = open(NAME, O_RDWR);
        if (_open == -1) {
            perror("open() Error");
            exit(-1);
        }

        strcpy(_buffer, "Kiss Mate NXYCH1\0");
        write(_open, _buffer, SIZE);

        close(_open);

        exit(1);
    } else { // az a parent
        printf("%d: I'm a parent [1.]\n", getpid());

        _fifo = mkfifo(NAME, 0100);
        if (_fifo == -1) {
            perror("mkfifo() Error");
            exit(-1);
        }
        close(_fifo);

        wait(NULL);

        printf("%d: I'm a parent [2.]\n", getpid());

        _open = open(NAME, O_RDWR);
        if (_open == -1) {
            perror("open()");
            exit(-1);
        }

        printf("0\n");

        while (read(_fifo, _buffer, SIZE) > 0) {
            printf("%s", _buffer);                              // olvasás
        }
        printf("\n");

        close(_open);
        unlink(NAME);

        exit(1);
    }
    return 0;
}
```

3. Feladat:

Írjon C nyelvű programot, amelyik kill() seg.-vel SIGALRM-et küld egy argumentumként megadott PID-u processznek, egy másik futó program a SIGALRM-hez rendeljen egy fv.-t amely kiírja pl.neptunkodot, továbbá pause() fv.-el blokkolódjon, majd kibillenés után jelezze, hogy kibillent és terminálódjon. Mentés. neptunkod_gyak9_3.c

Start here | nxych1_gyak_3.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>

int kill(pid_t pid, int sig);

int main(int argc, char **argv)
{
    if(argc != 2)
    {
        printf("Hasznalat: ./nxych1_gyak9_3 PID\n");
        return 1;
    }

    pid_t pid = (pid_t)atoi(argv[1]);
    kill(pid, SIGALRM);
    return 0;
}
```

Start here | nxych1_gyak_3_2.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>

void AlarmHandler(int sig);

int main(void)
{
    if (signal(SIGALRM, AlarmHandler) == SIG_ERR)
    {
        printf("Nem sikerult handlert allitani a(z) \"SIGALRM\" jelre!\n");
        return 1;
    }

    pause();
    return 0;
}

void AlarmHandler(int sig)
{
    printf("NXYCH1\n   Mostantol nem blokkol !\n");
    exit(1);
}
```

4. Feladat:
   Írjon C nyelvű programot, amelyik a SIGTERM-hez hozzárendel egy fv-t., amelyik kiírja az int paraméter értéket, majd végtelen ciklusban fusson, 3 sec-ig állandóan blokkolódva elindítás után egy másik shell-ben kill paranccsal (SIGTERM) próbálja terminálni, majd SIGKILL-el. Mentés. neptunkod_gyak9_4.c

Start here | nxych1_gyak_4.c

```c
#include <stdio.h>
#include <stdlib.h>

#include <unistd.h>
#include <signal.h>

void TerminationHandler(int sig);

int main(void)
{
    if (signal(SIGTERM, TerminationHandler) == SIG_ERR)
    {
        printf("Nem sikerult a handlert allitani a \"SIGTERM\" jelre!\n");
        return 0;
    }

    while(1)
    {
        printf("Varakozom...\n");
        sleep(3);
    }

    return 0;
}

void TerminationHandler(int sig)
{
    signal(sig, SIG_IGN);
    printf("SIGTERM signal: %d\n", sig);
}

```