

دانشگاه علم و صنعت

دانشکده مهندسی کامپیوتر

درجه تحصیلی: کارشناسی

گزارشکار تکلیف2 OS

گردآورنده:

پرنیان شاکریان - 99400064

استاد:

دکتر انتظاری

سال تحصیلی: اسفند 1401

تمام کتابخانه ها و headerهای مورد نیاز ما برای تمرین 2 به طور کلی و برای آسانی کار اینجا نوشته و اشاره شده است.

#include <assert.h>

#include <stdio.h>

#include <pthread.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <sys/wait.h>

#include <string.h>

برای انجام این سوال از لینک و سایت های زیر کمک گرفته ام.

<https://www.geeksforgeeks.org/create-directoryfolder-cc-program/>

<https://stackoverflow.com/questions/49881924/use-of-fork-to-create-folders>

<https://www.tutorialspoint.com/c_standard_library/c_function_system.htm>

<https://www.tutorialspoint.com/system-function-in-c-cplusplus>

<https://youtu.be/teu0_tEKJLw>

<https://youtu.be/7VOW4zkDZrQ>

سوال 4: برنامه ای بنویسید که که یک آرگیومنت بگیرد و با استفاده از fork و دستور mkdir یک فولدر به همان نام بسازد و درون اون فولدر با استفاده از دستور touch یک فایل تکست به اسم شماره دانشجویی خودتان بسازد و با استفاده از زبان C داخل اون فولدر چیزی رو به دلخواه write کند.

به طور خلاصه برنامه ما یک دایرکتوری جدید با نام مشخصی ایجاد می کند که به عنوان آرگومان خط فرمان ارسال می شود، سپس یک فایل جدید در دایرکتوری با نامی که در برنامه تعریف شده است ایجاد می کند و یک محتوای متنی مشخص را در فایل می نویسد. این برنامه از fork() برای ایجاد یک پردازش فرزند جدید استفاده می کند، که سپس دستور mkdir را با نام دایرکتوری جدید به عنوان آرگومان با استفاده از execvpاجرا می کند. فرآیند parent منتظر می ماند تا پردازش فرزند با فراخوانی تابعwaitpid اجرای mkdir را تمام کند و دوباره با استفاده ازfork یک پردازش فرزند جدید ایجاد می کند، که دستور touch را برای ایجاد یک فایل جدید با نام از پیش تعریف شده در دایرکتوری تازه ایجاد شده فرآیند والد دوباره منتظر می ماند تا پردازش فرزند به پایان برساند و سپس محتوای متن را با استفاده از توابع استاندارد C در فایل جدید ایجاد شده می نویسد.

شرح تکمیلی:

تابع main آرگومان‌ها را از خط فرمان دریافت می کند. بررسی می‌کند که آیا تعداد آرگومان‌ها کمتر از 2 باشد، به این معنی که نام پوشه‌ای ارائه نشده، و قبل از خروج از برنامه یک پیام خطا چاپ می‌کند. سه نشانگر کاراکتری را برای ذخیره folder\_name، text\_name و text\_content اعلام می‌کنیم. folder\_name به عنصر دوم آرایه آرگومان مقداردهی اولیه می شود که نام دایرکتوری جدیدی است که می خواهیم ایجاد کنیم. text\_name به رشته ثابت "99400064.txt" مقدار دهی اولیه می شود، که نام فایلی است که میخواهیم ایجاد کنیم. text\_content به رشته "Doing OS homework." مقداردهی اولیه می شود، که محتوایی است که می خواهیم در فایل ایجاد شده جدید بنویسیم.

دو آرایه از نشانگرهای کاراکتر، mkdir\_args و touch\_args را برای ذخیره آرگومان های دستوری که به ترتیب به دستورات 'mkdir' و 'touch' ارسال می شوند، اعلام می کند. mkdir\_args با mkdir، folder\_name و NULL مقداردهی اولیه می شود که به ترتیب آرگومان های دستور mkdir هستند. touch\_args با touch، filepath و NULL مقداردهی اولیه می شود که به ترتیب آرگومان های فرمان هستند. متغیر filepath با به هم پیوستن folder\_name، و text\_name با استفاده از تابع snprintf ساخته می شود.

با فورک یک پردازش فرزند جدید ایجاد می کند و شناسه فرآیند پردازش فرزند را به فرآیند والد برمی گرداند. شناسه فرآیند برگشتی در pid ذخیره می شود. فرآیند والد بررسی می‌کند که آیا شناسه فرآیند برگشتی مثبت است یا نه. اگر این باشد تابع waitpid را فراخوانی می‌کند تا منتظر بماند تا فرآیند فرزند اجرای دستور mkdir را قبل از رفتن به دستورالعمل‌های بعدی به پایان برساند. پس از ایجاد دایرکتوری، با اجرای دوباره fork یک پردازش فرزند جدید ایجاد می کند و شناسه فرآیند خود را در touch\_pid ذخیره می کند. پردازش فرزند جدید بررسی می کند که آیا touch\_pid منفی است، به این معنی که خطایی وجود داشته است، و قبل از خروج یک پیام خطا چاپ می کند. اگر منفی نباشد، بررسی می‌کند که touch\_pid برابر با 0 است یا خیر، یعنی فرآیند فرزند است. در آن صورت، execvp را برای اجرای فرمان touch با آرگومان های فرمان ذخیره شده در touch\_args و به دنبال آن 'exit(0)' برای خاتمه دادن به فرآیند فرزند فراخوانی می کند. فرآیند والد منتظر می ماند تا پردازش فرزند دوم اجرای touch را به پایان برساند. هنگامی که فایل ایجاد شد، با استفاده از تابع fopen برای نوشتن روی فایل، نشانگر فایل را باز می کند. سپس، محتوای متن را با استفاده از fputs روی فایل می‌نویسد و با استفاده از fclose فایل را می‌بندد. اگر فورک شکست خورد، برنامه قبل از خروج یک پیام خطا چاپ می کند.

خروجی:

