



دانشگاه علم و صنعت

دانشکده مهندسی کامپیوتر

درجه تحصیلی: کارشناسی

گزارشکار تکلیف OS 2

گردآورنده:

پرnian شاکریان - 99400064

استاد:

دکتر انتظاری

سال تحصیلی: اسفند ۱۴۰۱

تمام کتابخانه ها و headerهای مورد نیاز ما برای تمرین ۲ 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 ۴: برنامه ای بنویسید که یک آرگيومنت 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 `execvp` اجرا می کند. فرآیند `parent` منتظر می ماند تا پردازش فرزند با فراخوانی تابع `waitpid` اجرای `mkdir` را تمام کند و دوباره با استفاده از `fork` 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.

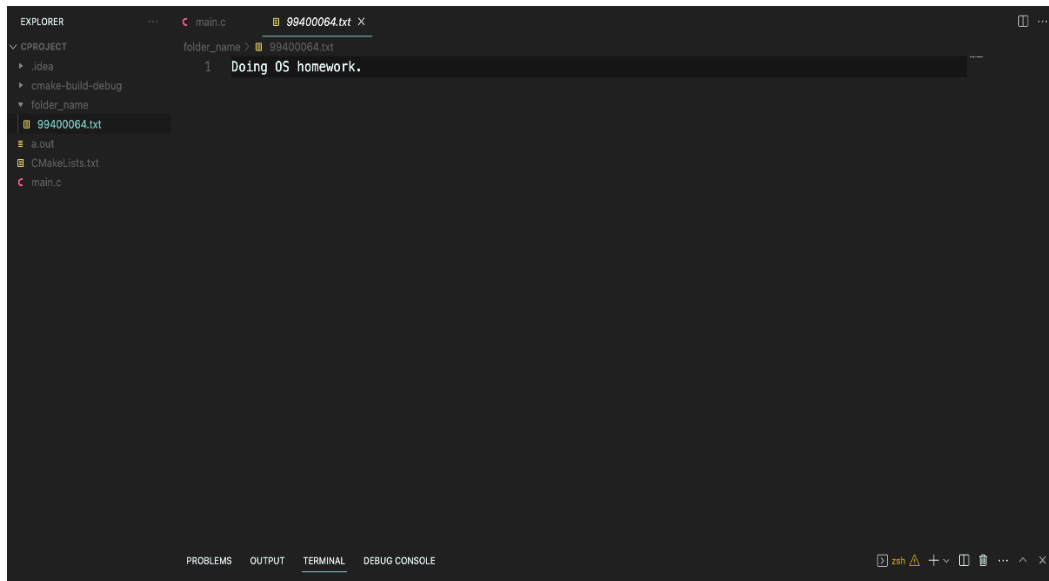
شرح تکمیلی:

تابع `main` آرگومان ها را از خط فرمان دریافت می کند. بررسی می کند که آیا تعداد آرگومان ها کمتر از ۲ باشد، به این معنی که نام پوشه ای ارائه نشده، و قبل از خروج از برنامه یک پیام خطا چاپ می کند. سه نشانگر کاراکتری را برای ذخیره `folder_name`، `text_name` و `text_content` اعلام می کنیم. `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 `"txt.۹۹۴۰۰۰۶۴"` مقداردهی اولیه می شود، که نام فایلی است که می خواهیم 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 می کند که آیا شناسه فرآیند برگشتی مثبت است یا نه. اگر این باشد تابع `waitpid` را فراخوانی می کند تا منتظر بماند تا فرآیند فرزند اجرای دستور `mkdir` را قبل از رفتن به دستورالعمل های بعدی به پایان برساند. پس از ایجاد دایرکتوری، با اجرای دوباره یک `fork` 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 می کند که `touch_pid` برابر با ۰ 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 می نویسد و با استفاده از `fclose` فایل را می بندد. اگر فورک شکست خورد، برنامه قبل از خروج یک پیام خطا چاپ می کند.

خروجی:



The screenshot shows a CMake IDE interface. On the left, the 'EXPLORER' panel displays a project structure under 'CPROJECT'. The structure includes a 'folder_name' directory containing '99400064.txt', 'a.out', 'CMakeLists.txt', and 'main.c'. The 'main.c' file is selected, and its content is displayed in the main editor area. The code in 'main.c' is as follows:

```
1 Doing OS homework.
```

The bottom of the IDE shows a status bar with tabs for 'PROBLEMS', 'OUTPUT', 'TERMINAL', and 'DEBUG CONSOLE'. The 'TERMINAL' tab is active, showing a 'zsh' prompt.