دانشگاه علم و صنعت

دانشکده مهندسی کامپیوتر

درجه تحصیلی: کارشناسی

گزارشکار تکلیف2 OS

گردآورنده:

پرنیان شاکریان - 99400064

استاد:

دکتر انتظاری

سال تحصیلی: اسفند ۱۴۰۱

خلاصه:

در تکلیف دوم سیستم عامل قصد داریم از Process و Thread در حل تمارین استفاده کنیم. اگر بخواهیم به طور خلاصه در مورد ماهیت کدها توضیح دهیم Process، پردازه مجموعه‌ای از کدها، حافظه، داده و سایر منابع است. Thread توالی از کدها به حساب می‌آید که در داخل محدوده یک پردازه اجرا می‌شود. هر رشته می تواند بخشی از یک وظیفه را اجرا کند (به عنوان مثال، جمع کردن عناصر یک آرایه)، یا می تواند همان کار را برای client مختلف در client-server اختصاص دهد.

شرح کلی:

۱. اضافه کردن کتابخانه‌ها(Include the header file)
۲. هر رشته دارای یک شی از نوع pthread_t مرتبط با آن است که شناسه آن را می گوید. (نمی تواند توسط چندین رشته به طور همزمان استفاده شود)
۳. یک رشته ایجاد می شود و با استفاده از تابع ()pthread_create شروع می شود و نیازمند چهار پارامتر است (id, Attributes, Starting routine, Arguments)
۴. () pthread_exit برای خروج از thread استفاده می شود.(معمولاً در پایان روتین شروع نوشته می شود.) اگر مقداری پس از پایان توسط یک رشته برگردانده شود، مرجع آن به عنوان یک آرگومان ارسال می شود.
۵. parent با استفاده از ()pthread_join منتظر یک child می‌شود. دو پارامتر این تابع عبارتند از Thread ID و Reference to return value

نکته: (نوع برگشتی شروع و آرگومان آن معمولاً بر روی *void تنظیم می شود.)

تمام کتابخانه ها و headerهای مورد نیاز ما برای تمرین ۲ به طور کلی و برای آســـانی 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>

#include "stdlib.h"

#include "stdio.h"

#include "unistd.h"

#include "string.h"

#include "pthread.h"
```

برای ساخت makefile ها از لینک های زیر کمک گرفته ام.

https://youtu.be/bPMDEdjKBWA

https://youtu.be/_r7i5X0rXJk

سوال ۱: تابع توان، تابعی پیاده سازی کنید که با O(log n) عدد x را به توان y برساند.

در ابتدا کتابخانه‌های مورد نظر خود را include خواهیم کرد. (کتابخانه های مورد نیاز ما برای این تمرین در اول داک نوشته شده است)

حالت کلیه کد روشی برای به توان رسوندن عدد این است که جواب x به توان y :

اگر y زوج باشد میشود x^(y/2) * x^(y/2)

اگر هم y فرد باشد میشود x^(y/2) * x^(y/2+1)

حالا اگر بخواهیم فرمول فوق را محاسبه کنیم، باید برای هر بخش محاسبه یک ترد بزنیم. برای مثال وقتی روی تابعی هستیم که a به توان b رو حساب میکنیم، یه ترد خواهیم زد که a^(b/2) را حساب کند و جواب را میذاریم res1 و یک ترد دیگر هم میزنیم که a^(b/2+b%2) را حساب کند و جواب را میذاریم res2. وقتی ترد ها حساب شدن و جوابشان محاسبه شد، جوابی که تابع ما برمیگرداند میشود res1*res2.

توجه شود که فقط دو تا حالت پایه هم دارد اینکه:

اگر y=0 باشد جواب میشود ۱

اگر y=1 باشد جواب میشود x

(خلاصه کلی این قسمت از تابع این است که بررسی می کند آیا توان ۰ یا ۱ است. اگر توان بزرگتر از ۱ باشد، تابع با استفاده از تقسیم اعداد صحیح، توان را به دو قسمت تقسیم می کند و دو نخ فرزند برای محاسبه توان هر قسمت ایجاد می کند. سپس منتظر می ماند تا هر دو رشته محاسبات خود را به پایان برسانند و نتایج خود را برای به دست آوردن پاسخ نهایی ضرب می کند.)

روش ترد به گونه‌ای است که گاهی خروجی گرفتن از آن ممکن است سخت باشد. همچنین ورودی تابعی که به ترد میدهیم، باید یک عدد باشد. (نمیتوانیم چند پارامتر ورودی بدیم) به همین دلیل در ابتدا یک استراکت درست میکنیم که x و y وans را در بردارد که x وy ورودی های تابع ترد هستند و ans جوابش میباشد. (ساختار custom_data ورودی‌های لازم را برای محاسبه توان، از جمله عدد پایه x، توان y و نتیجه محاسبه‌شده نگه می‌دارد.) تابع custom_power یک pointer به داده های ورودی دریافت می کند، مقادیر x و y را استخراج کرده و توان را به صورت بازگشتی محاسبه می کند.

در تابع main، کاربر مقادیر x و y را وارد کرده و داده های ورودی را برای محاسبه توان ایجاد می کند. سپس یک رشته جدید برای فراخوانی تابع custom_power و محاسبه نتیجه ایجاد می شود. در نهایت پس از اتمام اجرای thread، نتیجه محاسبه شده در کنسول چاپ می شود. کد نهایی خود را ران میکنیم، سپس مقدار ورودی اول و دوم خود را وارد کرده و در اخر enter میزنیم (میتوان ورودی داخواه داد).

خروجی‌های زیر را برای مثال اول و دوم در سایت کوئرا خواهد داشت.

main.c

Run

Output

```c
int main() {
    double x, ans = 0;
    int y;

    printf("x = ");
    scanf("%lf", &x);

    printf("y = ");
    scanf("%d", &y);

    if (y == 0){
        ans = x;
    }else{
        struct custom_data function_input = {x, y, 0};

        pthread_t thread;
        pthread_create(&thread, NULL, custom_power, &function_input);
        pthread_join(thread, NULL);

        ans = function_input.ans;
    }

    printf("%lf", ans);

    return 0;
}
```

```
/tmp/EUGWgPjhhN.o
x = 2.1

y = 10
1667.988098
```