**实验1.2线程**

**步骤一：**

#include<stdio.h>

#include<pthread.h>

#include<unistd.h>

#include<stdlib.h>

#include<sys/types.h>

#include<sys/wait.h>

int value=0;

void \*runner(void \*param); /\* the thread \*/

void \*runner2(void \*param); /\* the thread \*/

int main()

{

pthread\_t tid[10];

pthread\_attr\_t attr;

pthread\_attr\_init(&attr);

if(pthread\_create(&tid[0], &attr, runner, NULL) )

{

perror("Failed to create thread1");

exit(1);

}

if(pthread\_create(&tid[1], &attr, runner2, NULL))

{

perror("Failed to create thread2");

exit(1);

}

pthread\_join(tid[0], NULL);

pthread\_join(tid[1], NULL);

printf("variable result:%d", value);

}

void \*runner(void \*param)

{

printf("thread1 creat success!!!\n");

for(int i=0;i<100000 ;i++)

value++;

pthread\_exit(0);

}

void \*runner2(void \*param)

{

printf("thread2 creat success!!!\n");

for(int i=0;i<100000 ;i++)

value--;

pthread\_exit(0);

}

**步骤二：**

#include<stdio.h>

#include<pthread.h>

#include<unistd.h>

#include<sys/types.h>

#include<sys/wait.h>

#include<stdlib.h>

#include<semaphore.h>

sem\_t sem;

int value=0;

int s=1;

void \*runner(void \*param); /\* the thread \*/

void \*runner2(void \*param); /\* the thread \*/

int main()

{

if(sem\_init(&sem,0,1))

{

perror("Failed to initialize semaphore");

exit(1);}

pthread\_t tid[10];

pthread\_attr\_t attr;

pthread\_attr\_init(&attr);

if(pthread\_create(&tid[0], &attr, runner, NULL) )

{

perror("Failed to create thread1");

exit(1);}

if(pthread\_create(&tid[1], &attr, runner2, NULL))

{

perror("Failed to create thread2");

exit(1);}

pthread\_join(tid[0], NULL);

pthread\_join(tid[1], NULL);

sem\_destroy(&sem);

printf("variable result:%d", value);

}

void \*runner(void \*param)

{

printf("thread1 creat success!!!\n");

sem\_wait(&sem);

for(int i=0;i<100000 ;i++)

{

value++;

}

sem\_post(&sem);

pthread\_exit(0);

}

void \*runner2(void \*param)

{

printf("thread2 creat success!!!\n");

sem\_wait(&sem);

for(int i=0;i<100000 ;i++)

{

value--;

}

sem\_post(&sem);

pthread\_exit(0);

}

**步骤三：**

#include<stdio.h>

#include<pthread.h>

#include<unistd.h>

#include<sys/types.h>

#include<sys/wait.h>

#include <sys/syscall.h>

#include<stdlib.h>

#include<semaphore.h>

sem\_t sem;

int value=0;

int s=1;

void \*runner(void \*param); /\* the thread \*/

void \*runner2(void \*param); /\* the thread \*/

int main()

{

if(sem\_init(&sem,0,1))

{

perror("Failed to initialize semaphore");

exit(1);}

pthread\_t tid[10];

pthread\_attr\_t attr;

pthread\_attr\_init(&attr);

if(pthread\_create(&tid[0], &attr, runner, NULL) )

{

perror("Failed to create thread1");

exit(1);}

if(pthread\_create(&tid[1], &attr, runner2, NULL))

{

perror("Failed to create thread2");

exit(1);}

pthread\_join(tid[0], NULL);

pthread\_join(tid[1], NULL);

printf("variable result:%d", value);

}

void \*runner(void \*param)

{

printf("thread1 creat success!!!\n");

printf("thread1 tid=%lu,pid=%d\n",(unsigned long)pthread\_self(),getpid());

//system("./system\_call");

execl("/bin/sh", "sh", "-c", "./system\_call", (char \*)0);

printf("thread1 systemcall return\n");

pthread\_exit(0);

}

void \*runner2(void \*param)

{

printf("thread2 creat success!!!\n");

printf("thread2 tid=%lu,pid=%d\n",(unsigned long)pthread\_self(),getpid());

///system("./system\_call");

execl("/bin/sh", "sh", "-c", "./system\_call", (char \*)0);

printf("thread2 systemcall return\n");

pthread\_exit(0);

}

**system\_call.c源代码：**

#include<stdio.h>

#include <pthread.h>

#include <unistd.h>

#include<sys/types.h>

#include<sys/wait.h>

#include<stdlib.h>

int main()

{

int pid=getpid();

printf("system\_call PID:%d\n",pid);

return 0;}