Homework Assignment #2

Multi-Thread Programming: Matrix Computation

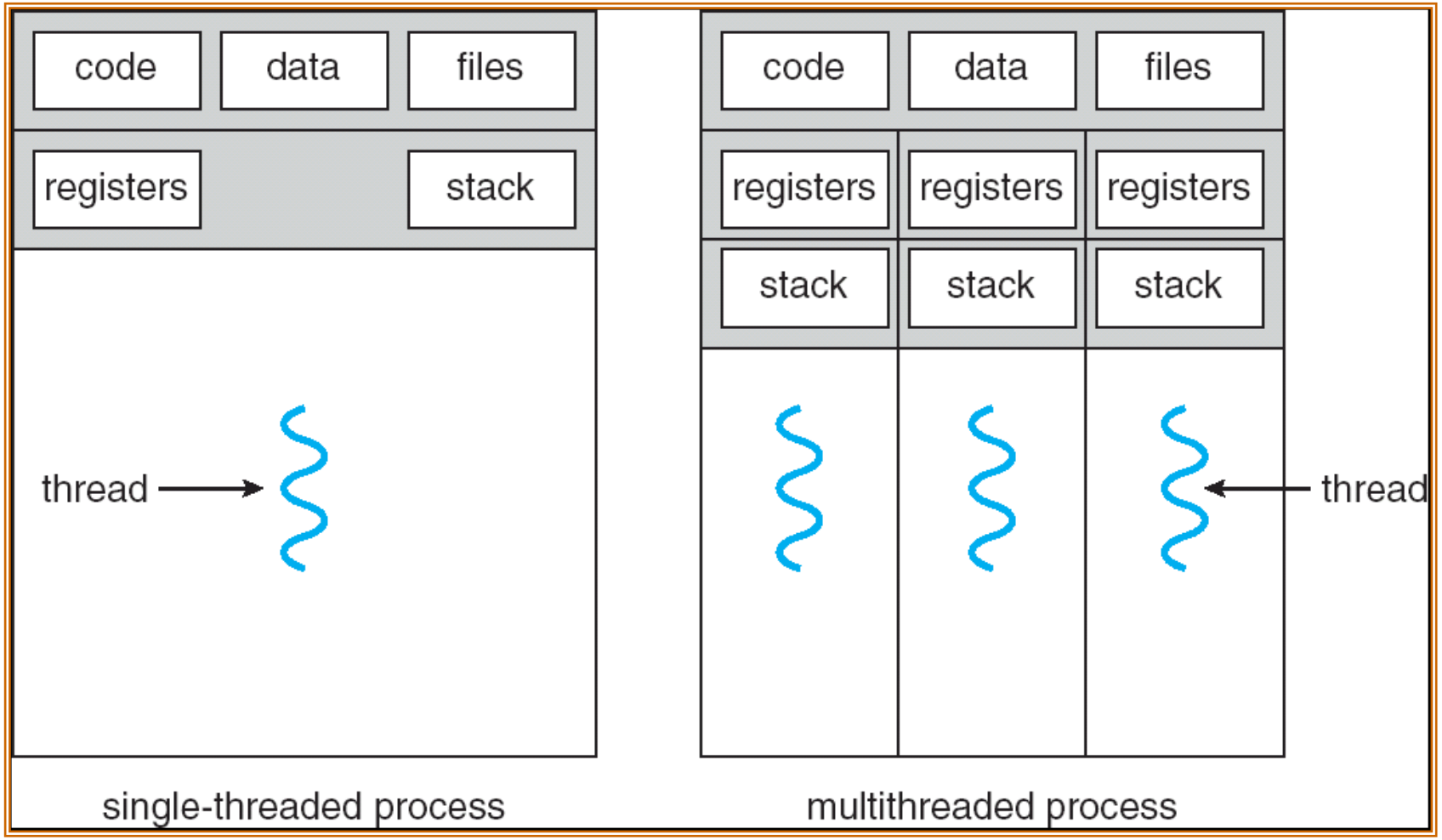
Outline

- **Thread**
- **Homework assignment 2**

Outline

- **Thread**
- **Homework assignment 2**

What is Thread?



Thread

- A thread is a flow of control within a process

- **Describe:**
- Thread *shares* with other threads in the same process its *code section*, *data section*, and other OS resources.

Thread API

- pthread_create()
  - The **pthread_create**() function starts a new thread in the calling process.

```
#include <pthread.h>
```

```
int pthread_create(pthread_t *thread, const pthread_attr_t *attr, void *(*start_routine) (void *), void *arg);
```

```
pthread_create(&thread1, NULL, &function, NULL);
```

Thread API(cont.)

- pthread_exit()
  - Exit a thread without exiting a process

```
#include <pthread.h>
```

```
void pthread_exit(void *retval);
```

```
pthread_exit(NULL);
```

Thread API(cont.)

- pthread_join()
  - Wait for a thread

```
#include <pthread.h>
```

```
int pthread_join(pthread_t thread, void **retval);
```

```
int pthread_join(thread1, NULL);
```

Example

```c
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
void* run(){
        int c=0;
        for(int i=0;i<10000000;i++)
                c++;
        printf("%d\n",c);
        pthread_exit(NULL);
}
int main(){
        pthread_t t1,t2;
        pthread_create(&t1,NULL,&run,NULL);
        pthread_create(&t2,NULL,&run,NULL);
        pthread_join(t1,NULL);
        pthread_join(t2,NULL);
        return 0;
}
```

Example(cont.)

```
andrew@andrew-ubuntu:~$ gcc -o test1 test1.c -pthread
andrew@andrew-ubuntu:~$ ./test1
1000000000
1000000000
```

Pass Argument During Thread Creation

```
int pthread_create(pthread_t *thread, const pthread_attr_t
*attr, void *(*start_routine) (void *), void *arg);
```

```
pthread_create(&thread, NULL, &run,(void*)&tmp)
```

```
void* run(void *argu){
        int n=*(int*)(argu);
        print(”%d”,n);
}
```

Example

```c
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
void* run(void *argu){
        int n=*(int*)(argu);
        printf("%d\n",n);
}
int main(){
        pthread_t t;
        int tmp=5;
        pthread_create(&t,NULL,&run,(void*)&tmp);
        pthread_join(t,NULL);
        return 0;
}
```

Example(cont.)

```
andrew@andrew-ubuntu:~$ gcc -o test1 test1.c -pthread
andrew@andrew-ubuntu:~$ ./test1
5
```

Outline

- **Thread**
- **Homework assignment2**

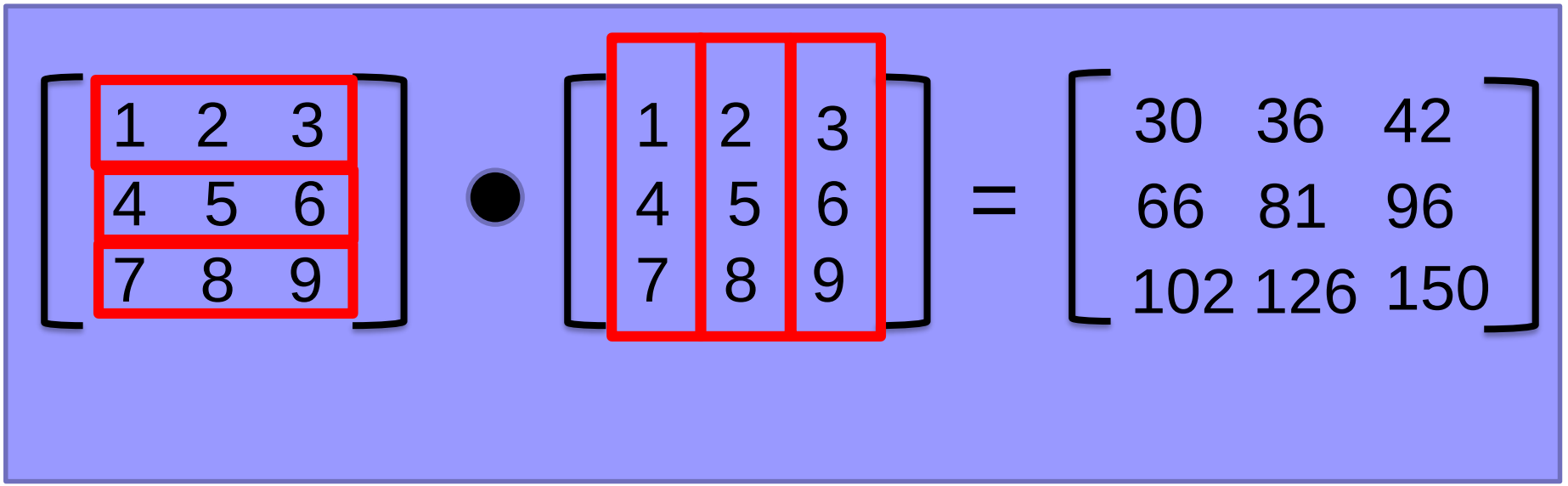
Homework 2:
Create Thread for Matrix Computation

- Key in an integer *n* as matrix size
- Key in two *n**n matrix C,D

1. For matrix C,D, you can key in by yourself. But each element needs to be an integer.

- Perform matrix computation: C*D = E (output)
  - Create multiple threads to compute the result concurrently
  - The number of created threads depends on the matrix dimension, i.e., *n* threads are created.
  - Thus, *i*-th thread calculates the result of the *i*-th row in the matrix ***E***.

Example



Result

```
andrew@andrew-ubuntu:/media/psf/Dropbox$ ./hw2
matrix size:
4*4
matrix a
1 2 3 4
5 6 7 8
9 8 7 6
5 4 3 2
matrix b
1 2 3 44
5 6 7 8
4 5 3 22
3 4 5 6
matrix C:
35 45 46 150
87 113 118 470
95 125 134 650
43 57 62 330
```

Turn in

- Deadline

  2020/12/3 PM.11:59:59
- Upload to iLearning
- File name
  - HW2_ID.zip (e.g. HW2_4106056000.zip)
    - Source code
      - .c file
    - Word(explain your code and post a screenshot of the result)
- If you don’t hand in your homework on time, your score will be deducted 10 points every day.

TA

- Name：Sheng-Ying Huang
- Email：g109056252@mail.nchu.edu.tw
- Lab：OSLab (1001)