

OS HW2 Programming exercises

資工三 110590004 林奕廷

Environment

- OS: Ubuntu 22.04
- Kernel: Linux 5.15.153.1-microsoft-standard-WSL2
- Compiler: gcc 11.4.0

4.24

Commands

```
cd 4.24
gcc main.c -pthread -o main
./main
3000000
```

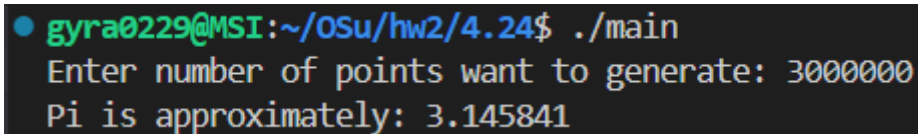
A terminal window showing the execution of the program in directory 4.24. The prompt is 'gyra0229@MSI:~/OSu/hw2/4.24\$'. The user enters './main'. The program prompts 'Enter number of points want to generate: 3000000' and then outputs 'Pi is approximately: 3.145841'.

Figure 1: The result for ./main in 4.24

4.27

Commands

```
cd 4.27
gcc main.c -pthread -o main
./main
20
```

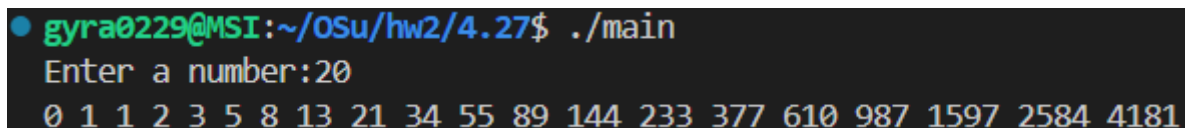
A terminal window showing the execution of the program in directory 4.27. The prompt is 'gyra0229@MSI:~/OSu/hw2/4.27\$'. The user enters './main'. The program prompts 'Enter a number:20' and then outputs a sequence of numbers: '0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181'.

Figure 2: The result for ./main in 4.27

6.33

```
1 #define MAX_RESOURCES 5
2 int available_resources = MAX_RESOURCES;
3 int decrease_count(int count) {
4     if (available_resources < count) {
5         return -1;
6     } else {
7         available_resources -= count;
8         return 0;
9     }
10 }
11 int increase_count(int count) {
12     available_resources += count;
13     return 0;
```

```
14 }  
15
```

- (a) The data involved race condition is `available_resources`.
- (b) The race condition occurs at line 4, 7, 12.
- (c) The revised version using mutex is as follows:

```
#include <pthread.h>  
#define MAX_RESOURCES 5  
int available_resources = MAX_RESOURCES;  
pthread_mutex_t mutex;  
int decrease_count(int count) {  
    pthread_mutex_lock(&mutex);  
    if (available_resources < count) {  
        pthread_mutex_unlock(&mutex);  
        return -1;  
    } else {  
        available_resources -= count;  
        pthread_mutex_unlock(&mutex);  
        return 0;  
    }  
}  
int increase_count(int count) {  
    pthread_mutex_lock(&mutex);  
    available_resources += count;  
    pthread_mutex_unlock(&mutex);  
    return 0;  
}
```