HW3 Programming Problems

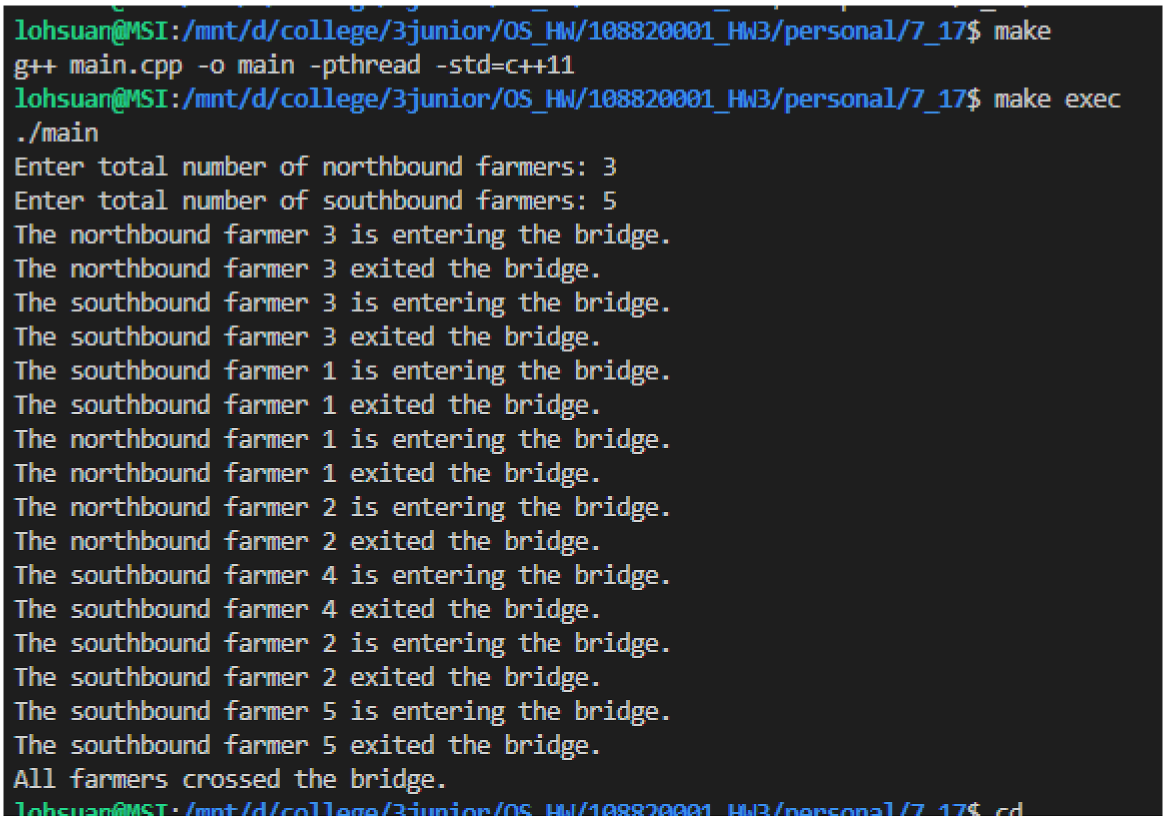
Q7.17

利用 multithread 實作南北邊的農夫要通過僅一台車寬度的橋，並避免 deadlock。

How to run

1. open terminal in this directory
2. enter `make` to compile
3. enter `make exec` or `./main` to execute
4. enter `make clean` to clean up (optional)

execution snapshot



```
lohsuan@MSI:/mnt/d/college/3junior/OS_HW/108820001_HW3/personal/7_17$ make
g++ main.cpp -o main -pthread -std=c++11
lohsuan@MSI:/mnt/d/college/3junior/OS_HW/108820001_HW3/personal/7_17$ make exec
./main
Enter total number of northbound farmers: 3
Enter total number of southbound farmers: 5
The northbound farmer 3 is entering the bridge.
The northbound farmer 3 exited the bridge.
The southbound farmer 3 is entering the bridge.
The southbound farmer 3 exited the bridge.
The southbound farmer 1 is entering the bridge.
The southbound farmer 1 exited the bridge.
The northbound farmer 1 is entering the bridge.
The northbound farmer 1 exited the bridge.
The northbound farmer 2 is entering the bridge.
The northbound farmer 2 exited the bridge.
The southbound farmer 4 is entering the bridge.
The southbound farmer 4 exited the bridge.
The southbound farmer 2 is entering the bridge.
The southbound farmer 2 exited the bridge.
The southbound farmer 5 is entering the bridge.
The southbound farmer 5 exited the bridge.
All farmers crossed the bridge.
lohsuan@MSI:/mnt/d/college/3junior/OS_HW/108820001_HW3/personal/7_17$ cd
```

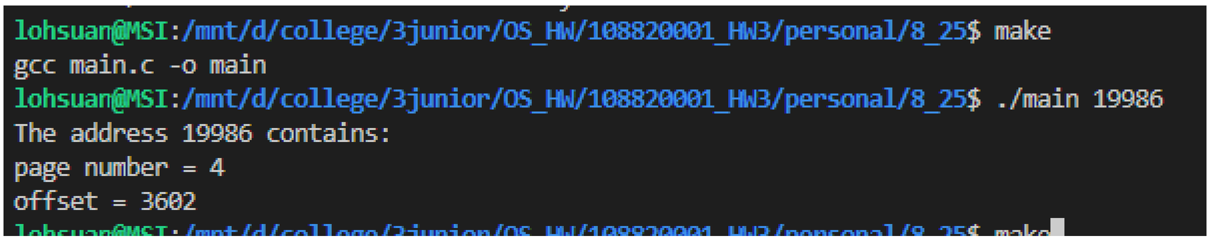
Q8.25

計算 32-bit virtual address, 4-KB page size virtual memory 對應的 page number 及 offset

How to run

1. open terminal in this directory
2. enter `make` to compile
3. enter `./main <address you want to check>` to execute
4. enter `make clean` to clean up (optional)

execution snapshot



```
lohsuan@MSI:/mnt/d/college/3junior/OS_HW/108820001_HW3/personal/8_25$ make
gcc main.c -o main
lohsuan@MSI:/mnt/d/college/3junior/OS_HW/108820001_HW3/personal/8_25$ ./main 19986
The address 19986 contains:
page number = 4
offset = 3602
lohsuan@MSI:/mnt/d/college/3junior/OS_HW/108820001_HW3/personal/8_25$ make
```

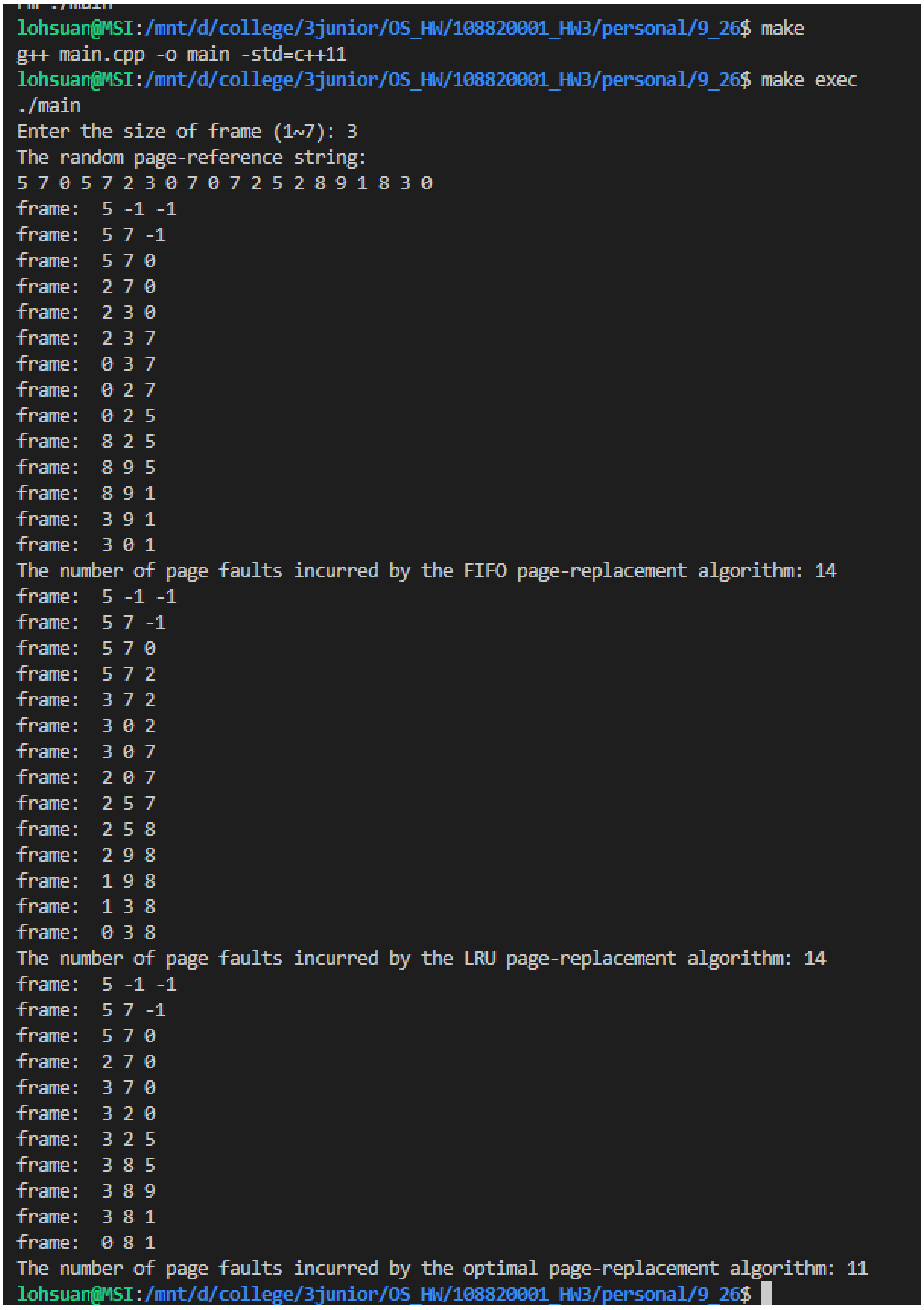
Q9.26

先產生隨機的 20 bit page-reference string，
再分別以 FIFO、LRU 和 optimal page-replacement algorithm 計算各自的 page fault。

How to run

1. open terminal in this directory
2. enter `make` to compile
3. enter `make exec` or `./main` to execute
4. enter `make clean` to clean up (optional)

execution snapshot



```
lohsuan@MSI:/mnt/d/college/3junior/OS_HW/108820001_HW3/personal/9_26$ make
g++ main.cpp -o main -std=c++11
lohsuan@MSI:/mnt/d/college/3junior/OS_HW/108820001_HW3/personal/9_26$ make exec
./main
Enter the size of frame (1~7): 3
The random page-reference string:
5 7 0 5 7 2 3 0 7 0 7 2 5 2 8 9 1 8 3 0
frame:  5 -1 -1
frame:  5 7 -1
frame:  5 7 0
frame:  2 7 0
frame:  2 3 0
frame:  2 3 7
frame:  0 3 7
frame:  0 2 7
frame:  0 2 5
frame:  8 2 5
frame:  8 9 5
frame:  8 9 1
frame:  3 9 1
frame:  3 0 1
The number of page faults incurred by the FIFO page-replacement algorithm: 14
frame:  5 -1 -1
frame:  5 7 -1
frame:  5 7 0
frame:  5 7 2
frame:  3 7 2
frame:  3 0 2
frame:  3 0 7
frame:  2 0 7
frame:  2 5 7
frame:  2 5 8
frame:  2 9 8
frame:  1 9 8
frame:  1 3 8
frame:  0 3 8
The number of page faults incurred by the LRU page-replacement algorithm: 14
frame:  5 -1 -1
frame:  5 7 -1
frame:  5 7 0
frame:  2 7 0
frame:  3 7 0
frame:  3 2 0
frame:  3 2 5
frame:  3 8 5
frame:  3 8 9
frame:  3 8 1
frame:  0 8 1
The number of page faults incurred by the optimal page-replacement algorithm: 11
lohsuan@MSI:/mnt/d/college/3junior/OS_HW/108820001_HW3/personal/9_26$
```