長庚大學109學年度第一學期作業系統期中測驗（滿分108）

系級: 姓名: 學號:

1. (8%) 在作業系統中，Interrupts是用來通知處理器來處理特殊事項的一個硬體機制與軟體技術，通常收到Interrupts後，作業系統會透過一連串的操作達成需要的結果。Interrupts又可以分為Software Interrupts與Hardware Interrupts，請分別舉出一項Software Interrupts與一項Hardware Interrupts？

Answer: Software Interrupts: signals, invalid memory access, division by zero, system calls, etc. (a correct answer +4%, a wrong answer -4%)
Hardware Interrupts: services requests of I/O devices, e.g., keyboards, Ethernet adapters, touch panels, etc. (a correct answer +4%, a wrong answer -4%)

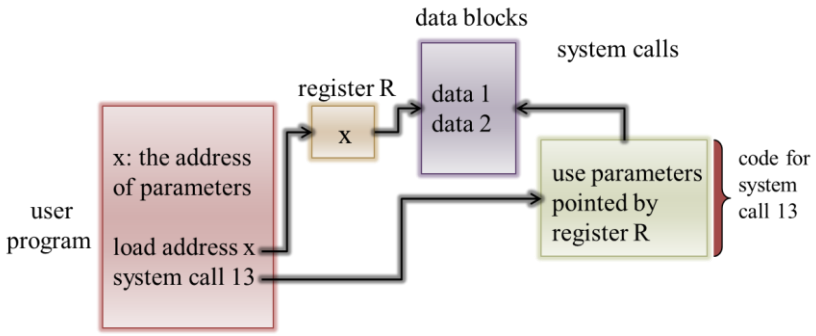
2. (8%) Real-Time Embedded Systems有許多常見的應用，譬如行車安全系統、手持式多媒體系統、居家或工廠自動控制系統。請說明Real-Time所指的意思為何？

Answer: Real-time means on-time instead of fast. Real-time systems have to get the correct (or qualified) computing results and have to make sure that the results can be derived on time.

3. (8%) 描述Application Programming Interface (API)、System Call、Operating System三者之間的關係。

Answer: System calls provide the routines for user applications to use the functions provided by operating systems (4%). The API of a programming language serves as a user-friendly link to system calls made available by the operating system (4%). Thus, most of the details of the operating-system interface are hidden from programmers by the API and are managed by run-time support libraries.

4. (8%) 呼叫System Call時如果應用程式需要傳遞參數給作業系統可以用以下三種媒介來傳遞：Registers、Stacks、Registers Pointing to Blocks。請說明Registers Pointing to Blocks是如何傳遞參數。下圖為投影片中的示意圖，可參考下圖作答。



Answer: The user application writes the parameters into data blocks and writes the pointer of the data blocks into a register before the user application invokes the system call. The system call then uses the pointer in the register to get the address of the data blocks and reads the parameters.

5. (8%) 在作業系統中請說明(a)Multiprogramming及(b)Time Sharing的定義。

Answer: (a) The operating system keeps several jobs in memory simultaneously (4%).
(b) Time sharing is a logical extension of multiprogramming, in which CPU switches jobs frequently so that users can interact with each job while it is running (4%).

6. (10%) 在Linux的作業系統環境中，每個一般的Process都會有Parent Process，所以當一個Process要被強制移除之前，如果其還有Child Process，就會產生Cascading Termination的現象。請問何謂Cascading Termination？

Answer: In some operating systems, a process can not be terminated when it has some child processes. Thus, before a process is forced to be terminated, all of its child processes have to be terminated. If any of its child processes further has some child processes, the grandchild processes have to be terminated as well, and so on and so forth.

7. (10%) 在實踐Inter-Process Communication時，其中一種做法是用Shared Memory的機制達成。在課本中提供了一個使用Shared Memory機制的範例：A Consumer-Producer Example。下圖是Consumer與Producer的程式碼。(a)請問在Consumer中 while (in = = out) ; 這行程式碼的目的為何？(b)在Producer中，while (((in + 1) % BUFFER SIZE) = = out) ; 這行程式碼需要Buffer的空間還剩下多少個以上(包含)才會允許Producer繼續進行寫入的動作？

```
Consumer:
while (true)
{
  while (in == out);
  /* do nothing and have to wait */
  next_consumed = buffer[out];
  out = (out + 1) % BUFFER_SIZE;
  … /* use the consumed item */
}
```

```
Producer:
while (true)
{
  … /* produce a new item */
  while (((in + 1) % BUFFER SIZE) == out) ;
  /* do nothing */
  buffer[in] = next_produced;
  in = (in + 1) % BUFFER SIZE;
}
```

Answer: (a) If the buffer is empty, the consumer should wait until the producer write any data item into the buffer.
(b) It requires that the remaining buffer space be no less than “2” data items.

8. (8%) 當我們在伺服器上設計網服務程式(如：網頁伺服器、FTP伺服器)，一般來說我們會用multiple threads而不是multiple processes來服務多位使用者。請問，相較之下使用multiple threads的優點為何？

Answer: Threads can share resources of a process, e.g., global data, binary code and opened files. Thus, it is much more efficient in terms of resource saving. Commutation among the threads of a process is easier than that among processes.

9. (8%) 請說明Thread Local Storage (TLS)的用途，並說明TLS與global variable有何不同。

Answer: Purpose: TLS allows each thread to have its own copy of data. (4%)
Difference: Global variables are visible for all function invocations in all threads of the process, but TLS is visible across function invocations in only a thread. (4%)

10. (8%) 請定義 (a) I/O-bound process 與 (b) CPU-bound process。

Answer: I/O-bound process – spends more time doing I/O than computations; many short CPU bursts.
CPU-bound process – spends more time doing computations; few very long CPU bursts.

11. (12%) 假設每次呼叫fork()都是成功的，請寫出以下程式在POSIX環境下執行後的輸出結果。

```
#include<sys/types.h>
#include<stdio.h>
#include<unistd.h>
int main()
{
        pid_t pid1, pid2, pid3;
        pid1 = fork();
        if (pid1 = = 0)
        {
                printf("AAA\n");
                pid2 = fork();
                if (pid2 > 0)
                {
                        wait(NULL);
                        printf("BBB\n");
                }
                else
                {
                        printf("CCC\n");
                }
```

```
    }
    else
    {
        wait(NULL);
        pid3 = fork();
        if (pid3 > 0)
        {
            wait(NULL);
            printf("EEE\n");
        }
        else
        {
            printf("FFF\n");
        }
        printf("GGG\n");
    }
    return 0;
}
```

Answer:

AAA
CCC
BBB
FFF
GGG
EEE
GGG

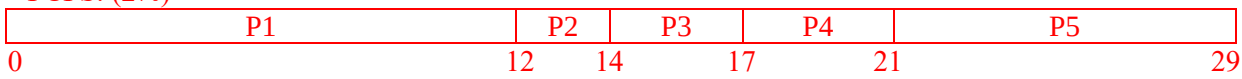
12. (12%) 考慮已經就緒的五個工作，依序為P1, P2, P3, P4, P5。使用三個排程演算法FCFS (First-Come, First-Served)、SJF (Shortest-Job-First)以及RR (Round Robin)來排程，而RR所使用的time quantum為4ms。(a)請畫下三個排程演算法的排程圖，(b)請分別算出三個排程演算法中每個工作的等待時間，若無算式一率不給分(算式可以只是簡單的加減法運算)，(c)請分別算出三個排程演算法的平均等待時間，若無算式一率不給分。

Process	Burst Time
P1	12 ms
P2	2 ms
P3	3 ms
P4	4 ms
P5	8 ms

Answer:

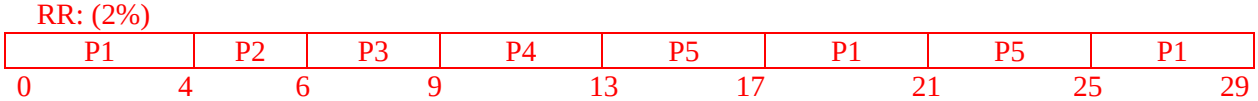
(1)

FCFS: (2%)



SJF: (2%)

RR: (2%)



(2)

FCFS: P1: 12-12=0, P2: 14-2=12, P3: 17-3=14, P4: 21-4=17, P5: 29-8=21 (1%)

SJF: P1: 29-12=17, P2: 2-2=0, P3: 5-3=2, P4: 9-4=5, P5: 17-8=9 (1%)

RR: P1: 29-12=17, P2: 6-2=4, P3: 9-3=6, P4: 13-4=9, P5: 25-8=17 (1%)

(3)

FCFS: (0+12+14+17+21)/5= 12.8 (1%)

SJF: (17+0+2+5+9)/5= 6.6 (1%)

RR: (17+4+6+9+17)/5= 10.6 (1%)