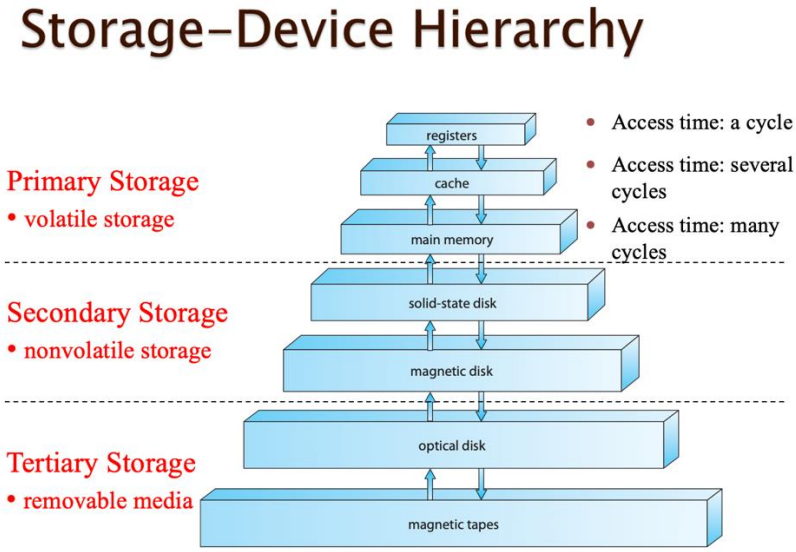
長庚大學111學年度第一學期作業系統期中測驗（滿分108）

系級: 姓名: 學號:

1. (8%) 下圖為儲存裝置與記憶體的分層架構，請問volatile這個形容詞在作業系統中，意指為何？



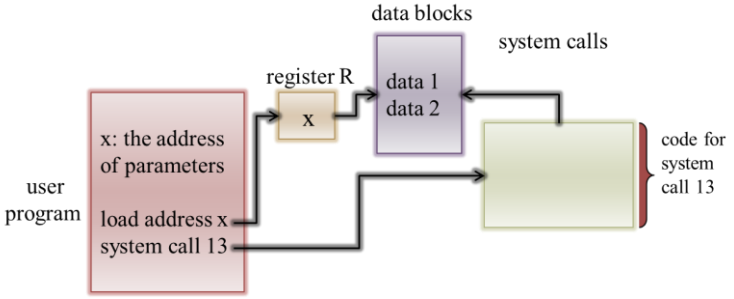
Answer: Volatile storage needs power supply to maintain the data, i.e., power outages could cause data loss, and nonvolatile storage can keep the data without power.

2. (8%) 在作業系統中請說明Multiprogramming (4%)及Time Sharing (4%)的定義。

Answer: Multiprogramming: The operating system keeps several jobs in memory simultaneously (4%).
Time Sharing: 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%).

3. (8%) 一般應用程式呼叫作業系統中提供的System Calls時，若需要傳遞參數給System Calls，有三種做法：Parameter Passing by Registers、Parameter Passing by Stacks、Parameter Passing by Registers Pointing to Blocks。請說明Parameter Passing by Registers Pointing to Blocks如何完成。

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.



4. (8%) Monolithic Kernel與Microkernel是兩種不同的作業系統設計方式，與Monolithic Kernel相比較之下，請舉出 (1)一個Microkernel的優點 (4%)、 (2)一個Microkernel的缺點 (4%)。

Answer: (1) Advantage: Microkernel is more modulized, and thus, it is more portable, reliable, and easy for extensions. (4%)
(2) Disadvantage: There are more inter-process communication (IPC) calls in Microkernel. Thus, the performance might be worse. (4%)

5. (8%) 在作業系統中有 Long-term Scheduler (或稱作 Job Scheduler)以及 Short-term Scheduler (或稱作 CPU scheduler)，請說明 (1) Long-term Scheduler (4%)與 (2) Short-term Scheduler (4%)的任務分別為何？

Answer: (1) Long-term scheduler selects which processes should be brought into the ready queue. (4%)
(2) Short-term scheduler selects which process should be executed next and allocates CPU. (4%)

6. (10%) 作業系統中在做Inter Process Communication (IPC)時有兩種方法：Message Passing and Shared Memory。兩相比較下，Message Passing對應用程式的開發者而言較容易使用；而需要大量資料傳輸與頻繁溝通時，妥善地使用Shared Memory將可以得到較好的效能。(1) 請說明為何Shared Memory效能較好 (4%)。以下為投影片中Consumer-and-Producer的範例程式，(2)請參考Producer的程式碼，完成Consumer中兩行空白的程式碼 (6%)。

Shared Memory- Producer

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

Shared Memory- Consumer

```
while (true)
{
  while (   LINE 1 for Question 2    );
  /* do nothing and have to wait */
  next_consumed = buffer[out];
  out =  LINE 2 for Question 2 ;
  … /* use the consumed item */
}
```

Answer: (1) If we use message passing, a serious of system calls must be invoked for sending each message. By using shared memory, multiple processes can directly access a shared memory area multiple times without invoking many system calls. (4%)
(2) Line 1: in == out (3%)
Line 2: (out + 1) % BUFFER_SIZE (3%)

7. (8%) 請說明 (1)Thread Local Storage (TLS)的用途，並 (2)說明TLS與global variable有何不同。

Answer: (1) Purpose: TLS allows each thread to have its own copy of data. (4%)
(2) Difference: Global variables are visible for all threads in the process, but TLS is visible in only a thread. (4%)

8. (8%) 請定義 (1) Kernel Thread與 (2) User Thread。

Answer: (1) Kernel threads are supported by the kernel and are the unit for OS CPU scheduling. (4%)
(2) Kernel threads are managed by the user-level thread library. (4%)

9. (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: (Only one correct reason is required)

1. 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.
2. Commutation among the threads of a process is easier than that among processes.

10. (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。

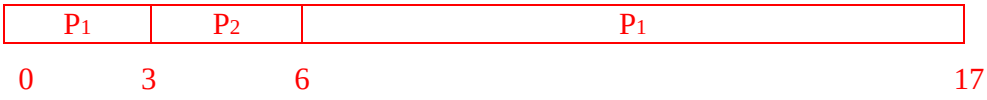
```
#include<sys/types.h>
#include<stdio.h>
#include<unistd.h>
int main()
{
        pid_t pid, pid2;
        pid = fork();
        if (pid == 0)
        {
                printf("Hello\n");
                pid2 = fork();
                if (pid2 != 0)
                {
                        wait(NULL);
                        printf("Hi\n");
                }
                else
                {
                        printf("Hola\n");
                }
        }
        else
        {
                wait(NULL);
                printf("Bonjour\n");
        }
        printf("Guten tag\n");
        return 0;
}
```

Answer: (-4 for each error)

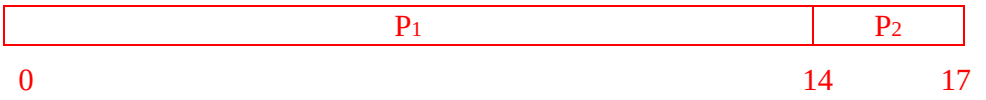
Hello
Hola
Guten tag
Hi
Guten tag
Bonjour
Guten tag

11. (12%) 有兩個工作P_1及P_2，所需的執行時間(Burst Time)分別是14與3，P_1於時間0到達，P_2於時間3到達，現在考慮兩個排程演算法Preemptive SJF以及Non-preemptive SJF。(1)請畫下兩個排程演算法的排程圖，(2)請分別算出兩個排程演算法的平均等待時間，若無算式一率不給分。

Answer:
(1) (6%)
Preemptive SJF:



Non-preemptive SJF:



(2) (6%) (一定要有算式才給分)
Preemptive SJF: ((17-14) + (6-3-3))/2=1.5
Non-preemptive SJF: ((14-14) + (17-3-3))/2=5.5

12. (8%) 使用Round Robin (RR) Scheduling 技術時，設定time quantum的大小是一件很重要的事，請說明 (a)當time quantum太大時會有什麼缺點？ (b)當time quantum太小時會有什麼缺點？

Answer: Too long: The RR scheduling will become the FCFS scheduling. Thus, the average waiting time could be long. (4%)
Too short: The overhead of context switches will be high. (4%)