2064

OLLSCOIL NA hÉIREANN, GAILLIMH
NATIONAL UNIVERSITY OF IRELAND, GALWAY

SEMESTER I EXAMINATIONS 2005/2006

SECOND YEAR COMPUTER SCIENCE [CS201]
OPERATING SYSTEMS and OBJECT ORIENTED PROGRAMMING in C++ [CS208]

Dr. Dave Johnson
Prof. J. Hinde
Dr. Claas Röver

Time allowed: **Two** hours.
Do **Questions 1 and 2** and **two** others

QUESTION 1. (a) What is an operating system and what are its main purposes? What is meant by a *multiprogramming system* and a *multitasking system*?

(b) Describe the *First-Come-First-Served* (FCFS), *Shortest-Job-First* (SJF), and *Round-Robin* (RR) algorithms for process scheduling. Give an advantage and disadvantage of each.

(c) Suppose at time $t = 0$, the following process have been submitted in the given order.

Process	Burst Time
P_1	11
P_2	8
P_3	15
P_4	6

For each of FCFS, SJF, and RR with a time quantum of $q = 4$, sketch the appropriate Gantt chart and calculate the average wait time for the processes.
(Assume that neither context switches nor terminating a process uses CPU time.)

(d) Explain the difference between preemptive and non-preemptive scheduling schemes. For each of FCFS, SJF and RR scheduling, state whether it is preemptive or not. Which hardware feature is required for RR scheduling?

QUESTION 2. As a C++ programmer on a software team you have been asked to develop a class for storing characters in a *stack* (LIFO queue). There should be methods for pushing characters onto the stack and popping them off, always one at a time. Furthermore, there should be a method for clearing the stack. The maximum number of characters on the stack is fixed at 25.

(a) Describe how you would design the class. State clearly which access modifiers, parameters and return values you would use. What does the constructor do?

(b) Give the C++ code for the class you designed in (a).

(c) For testing purposes, give a short C++ program that prompts the user for a string, and using a stack as temporary storage, prints that string backwards.

QUESTION 3. (a) Explain what a *process* is, in particular in contrast to a program. Name the different states a process can be in and explain their meaning. List four pieces of information which might be stored in a process control block.

(b) A child process can be created with the C function `fork()`. What values does it return to the parent and child processes? Do the child and parent processes run concurrently? Does the child process share the parent's memory resources?

(c) Write down one possible output of the following C program when it is executed with PID=4004 and PIDs are allocated in increasing order.

```
#include <stdio.h>
#include <unistd.h>
int main() {
  int num = 10;
  fork();
  printf("My PID=%d, num=%3d\n", getpid(), num);
  num *= 2;
  fork();
  printf("My PID=%d, num=%3d\n", getpid(), num);
  return 0;
}
```

QUESTION 4. (a) Explain the terms *physical address space* and *logical address space*. How does the OS deal with the situation where the physical address space is smaller than the logical address space?

(b) Describe what is meant by *paging* and *segmentation*. What is a page fault? List advantages and disadvantages of each method. What is a *cache memory* good for?

(c) Name and briefly describe three different policies for removing pages from memory when space is needed.

QUESTION 5. (a) Explain what a semaphore is and how it can be used to grant different threads mutually exclusive access to a resource.

(b) Give the C++ code for a program which forks a child process and then communicates its PID to the child via a pipe. The child should then print it to the screen.

(c) Assuming that all files are in the current directory, give the complete shell commands which you would use to

(i) rename the file `thesis.txt` to `master-thesis.txt`,

(ii) write a listing of the files in the current directory into the file `dir-list`,

(iii) set the permissions on the file `my-script.sh` so that you can read, write and execute it but others can only read it,

(iv) get a sorted output from your executable `random-gen` which normally prints a random sequence of integers onto the screen, one number per line.