Student ID & Name:

_____

Zhejiang University of Technology

Realtime Systems and Operating Systems

Examiners: Fuli Wu

Time allowed: 1.5 hours

Instructions:
There is a total of 60 marks available.

	Section A	Section B	Section C	Total
Score				

SECTION A

Attempt all questions. All questions in this section are worth 21 marks.

Question 1 *[3 marks]*

What are the three main purposes of an operating system?

Answer:

The three main purposes are:

1. To provide an environment for a computer user to execute programs on computer hardware in a convenient and efficient manner.
2. To allocate the separate resources of the computer as needed to solve the problem given. The allocation process should be as fair and efficient as possible.
3. As a control program it serves two major functions:(1) supervision of the execution of user programs to prevent errors and improper use of the computer, and (2) management of the operation and control of I/O devices.

Question 2 *[3 marks]*

Explain the difference between preemptive and non-preemptive scheduling.

Answer:

Preemptive scheduling allows a process to be interrupted during its execution, taking the CPU away and allocating it to another process. Non-preemptive scheduling ensures that a process relinquishes control of the CPU only when it finishes with its current CPU burst.

Question 3 *[5 marks]*

List at least five different types of computing environments.

Answer:

1. Time Sharing Computing Environment
2. Peer-to-Peer Computing Environment
3. Client Server Computing Environment
4. Web-Based Computing Environment
5. Cloud Computing Environment

or

1. Personal Computing Environment
2. Time Sharing Computing Environment
3. Client Server Computing Environment
4. Distributed Computing Environment
5. Cloud Computing Environment
6. Cluster Computing Environment

Question 4 *[5 marks]*

Of the following five forms of storage, rank them from fastest to slowest in terms of access time: (1) main memory, (2) solid state disk, (3) magnetic disk, (4) registers, (5) cache.

Answer:

(1) registers,
(2) cache,
(3) main memory,
(4) solid state disk,
(5) magnetic disk.

Question 5 *[5 marks]*

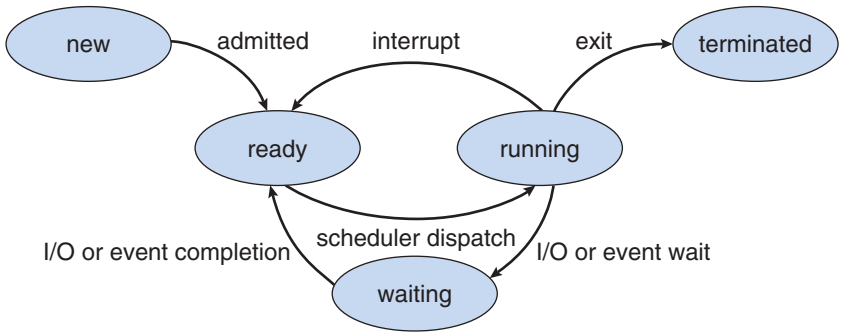
As a process executes, it changes state. The state of a process is defined in part by the current activity of that process. A process may be in one of five states. Please list the five states and illustrate the diagram of process state.

Answer:

A process may be in one of the following states:

1. New. The process is being created.
2. Running. Instructions are being executed.
3. Waiting. The process is waiting for some event to occur (such as an I/O completion or reception of a signal).
4. Ready. The process is waiting to be assigned to a processor.
5. Terminated. The process has finished execution.

The state diagram corresponding to these states is presented in the following.



SECTION B

Attempt all questions. All questions in this section are worth 21 marks.

Question 1 *[2 marks]*

Consider a logical address space of 1024 pages of 1024 bytes each, mapped onto a physical memory of 512 frames.

a. How many bits are there in the logical address?

b. How many bits are there in the physical address?

Answer:

```
a: 10 + 10 = 20bits
b: 9 + 10 = 19bits
```

Question 2 *[2 marks]*

Illustrate how a binary semaphore can be used to implement mutual exclusion among n processes.

Answer:

```
The n processes share a semaphore, mutex, initialized to 1.
Each process Pi is organized as follows:
    do {
         wait(mutex);        /* critical section */
         signal(mutex);     /* remainder section */
} while (true);
```

Question 3 *[2 marks]*

Including the initial parent process, how many processes are created by the program.

```
#include <stdio.h>
#include <unistd.h>
int main()
{
    int i;
    for (i = 0; i < 6; i++)
         fork();
    return 0;
}
```

Answer:

```
64
```

Question 4 *[4 marks]*

Consider the following code segment:

```
pid_t pid;
pid = fork();
if (pid == 0) { */ child process */
   fork();
   thread_create(…);
}
fork();
```

a. How many unique processes are created?

b. How many unique threads are created?

Answer:

this code segment creates a total of six processes and eight threads (two threads created by calls to thread_create() and six threads corresponding to the six single-threaded processes).

```
a: 8
b: 6
```

Question 5 *[5 marks]*

Using the program shown in Figure 3.35, explain what the output will be at lines X and Y.

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>

#define SIZE 5

int nums[SIZE] = {0, 4, 8, 12, 16};

int main()
{
    int i;
    pid t pid;
    pid = fork();
    if (pid == 0) {
        for (i = 0; i < SIZE; i++) {
            nums[i] *= (-i);
            printf("CHILD: %d ",nums[i]); /* LINE X */
        }
    }
    else if (pid > 0) {
        wait(NULL);
        for (i = 0; i < SIZE; i++)
            printf("PARENT: %d ",nums[i]); /* LINE Y */
    }
    return 0;
}
```

Answer:

X:

```
CHILD: 0
CHILD: -4
CHILD: -16
CHILD: -36
CHILD: -64
```

Y:

```
PARENT: 0
PARENT: 4
PARENT: 8
```

```
PARENT: 12
PARENT: 16
```

Question 6 *[6 marks]*

Suppose that the following processes arrive for execution at the times indicated. Each process will run for the amount of time listed. In answering the questions, use non preemptive scheduling, and base all decisions on the information you have at the time the decision must be made.

Process	Arrival Time	Burst Time
P1	0.0	7.6
P2	0.5	5.2
P3	1.0	2.2

1. What is the average turnaround time for these processes with the FCFS scheduling algorithm?

2. What is the average turnaround time for these processes with the SJF scheduling algorithm?

3. The SJF algorithm is supposed to improve performance but notice that we chose to run process P1 at time 0 because we did not know that two shorter processes would arrive soon. Compute what the average turnaround time will be if the CPU is left idle for the first 1 unit and then SJF scheduling is used. Remember that processes P1 and P2 are waiting during this idle time, so their waiting time may increase. This algorithm could be known as future-knowledge scheduling.

Answer:

1. *((7.6) + (7.6+5.2-0.5) + (7.6+5.2+2.2-1.0)) / 3 = 11.3*

2. *((7.6) + (7.6+2.2+5.3-0.5) + (7.6+2.2-1.0)) / 3 = 10.33*

3. *((2.2) + (2.2+5.2+1.0-0.5) + (2.2+5.2+7.6+1.0)) / 3 = 8.7*

SECTION C

Attempt all questions. All questions in this section are worth 18 marks.

Question 1 *[8 marks]*

The following processes are being scheduled using a preemptive, round- robin scheduling algorithm. Each process is assigned a numerical priority, with a higher number indicating a higher relative priority. In addition to the processes listed below, the system also has an idle task (which consumes no CPU resources and is identified as Pidle). This task has priority 0 and is scheduled whenever the system has no other available processes to run. The length of a time quantum is 10 units. If a process is preempted by a higher-priority process, the preempted process is placed at the end of the queue.

Process	Arrival Time	Burst	Priority
P1	0	20	42
P2	25	25	33
P3	30	25	33
P4	65	15	37
P5	100	10	8
P6	105	10	12

a. Show the scheduling order of the processes using a Gantt chart.
b. What is the turnaround time for each process?
c. What is the waiting time for each process?
d. What is the CPU utilization rate?

Answer:

P1.	P1.	P0.	P2.	P3.	P2.	P3.	P4.	P2.	P3.	P4.	P0.	P5.	P6.	P5
10.	10.	5.	10.	10.	10.	10.	10.	5.	5.	5.	10.	5.	10.	5

turnaround time:

P1: 20
P2: 80-25 = 55
P3: 85-30 = 55
P4: 90-65 = 25
P5: 20
P6: 10

waiting time:

P1: 0
P2: 55 – 25 = 30
P3: 55 – 30 = 25
P4: 25 – 15 = 10
P5: 20 – 10 = 10
P6: 0

CPU utilization: (20 + 25 + 25 + 15 + 10 + 10) / 120 = 105 / 120 = 87.5%

Question 2 *[10 marks]*

Suppose that a database is to be shared among several concurrent processes. Some of these processes may want only to read the database, whereas others may want to update (that is, to read and write) the database. We distinguish between these two types of processes by referring to the former as readers and to the latter as writers. To ensure that these difficulties do not arise, we require that reading and writing can't access the shared database simultaneously. This synchronization problem is referred to as the readers–writers' problem.

The simplest readers–writers' problem, requires that no reader be kept waiting unless a writer has already obtained permission to use the shared object. Please present a solution to the first readers–writers' problem. In the solution, the reader and writer processes share the following data structures:

```
semaphore write_binary_semaphor = 1;
semaphore reader_count_binary_semaphor = 1;
int reader_count = 0;
```

Please write the pseudo-code for the structure of a writer process and the structure of a reader process.

Answer:

writer process:

```
do {
    wait(write_binary_semaphor);
    ...
    /* writing is performed */
    ...
    signal(write_binary_semaphor);
} while (true);
```

reader process:

```
do {
    wait(reader_count_binary_semaphor);
    reader_count++;
    if (reader_count == 1)
       wait(write_binary_semaphor);
    signal(reader_count_binary_semaphor);
    ...
    /* reading is performed */
    ...
    wait(reader_count_binary_semaphor);
    reader_count--;
    if (reader_count == 0)
       signal(write_binary_semaphor);
    signal(reader_count_binary_semaphor);
} while (true);
```