Name: _____ Reg No.: _____ Signature: _____

CE 301 Operating Systems Fall 2021 Midterm Exam

Computer Engineering Department
Instructor: Dr. Rehan Ahmed

Date: December 2021 Max. Marks: 90 Time: 90 Minutes

Instructions (Please read carefully)

- This exam will assess your CLOs as per OBE. The CLOs are mentioned below.
- Write down your reg number on each page.
- Use of a calculator is allowed.
- This is a closed notes and closed book exam
- Answer without logic is illogical; provide logic and necessary steps for each answer.
- Answer all questions in the space provided. You may use the back side of the paper. **No extra sheets will be provided**.
- Use of unfair means/unauthorized material/unauthorized equipment will lead to immediate disqualification and disciplinary action.
- The points for each question/sub-question are given in braces

CLOs

1. **Understand** fundamental operating system abstractions such as processes, threads, files, semaphores, IPC abstractions, shared memory regions, etc.,
2. **Analyze** important algorithms and services provided by the operating systems: Scheduling, memory management, ls, cat, zip, etc
3. **Develop** operating systems services and add additional features to a simple operating system (xv6)
4. **Design** multi-threaded applications using the learnt parallel programming concepts

Q1(CLO -)	Q2(CLO1)	Q3(CLO2)	Q4(CLO2)	**Total**
/30	/15	/25	/20	/90

Reg No.:_____

Question 1 : (CLO–, -, PLO–) **[30]**

Objective: 15 MCQs. Expect to spend less than 15 minutes

1. A process which is currently executing on the CPU has the following state:

   a) Blocked
   b) Ready
   c) Running
   d) Zombie

2. A process which is waiting for an IO to occur has the following state:

   a) Blocked
   b) Ready
   c) Running
   d) Zombie

3. When a process is preempted, its state has the following transition:

   a) Blocked to Ready
   b) Ready to Running
   c) Running to Blocked
   d) Running to Ready

4. After executing the following instruction, the value of *rc* will be

```
1 int rc = fork()
```

   a) Parent rc = 0, Child rc = 0
   b) Parent rc = 0, Child rc = Child's PID
   c) Parent rc = 0, Child rc = Parent's PID
   d) Parent rc = Child's PID, Child rc = 0

5. Which of the following tasks does an OS **not** perform when a *trap* is called:

   a) Save all of the stack memory of the running process to a predefined memory location
   b) Read the trap table to get the location of the trap handler
   c) Save the general purpose registers of the running process
   d) execute the trap handler

6. The value of the timer used for non-cooperative scheduling is set by:

   a) BIOS
   b) User Program
   c) Operating System
   d) Hardwired

7. The *yield()* instruction is used in cooperative scheduling to:

   a) Signal the end of the program
   b) Force the processor into low power mode
   c) Give execution control back to the OS
   d) Generate a timer interrupt

8. The shortest time to completion first scheduling algorithm minimizes:

   a) Maximum turnaround time
   b) Maximum response time
   c) Average turnaround time
   d) Average response time

9. Round-robin scheduling algorithm minimizes:

a) Maximum turnaround time
b) Maximum response time
c) Average turnaround time
d) Average response time

10. In lottery scheduling, The number of tokens for process A and process B are $T_A = 80$, $T_B = 20$ respectively. Assuming that the computation time for process B is 10 seconds, determine the expected value of the turnaround time of process B. Assume that both processes arrive at time 0.

a) 10 sec
b) 12.5 sec
c) 50 sec
d) 20 sec

11. In a Multi Level Feedback Queue scheduling, which mechanism is used to avoid starvation:

a) Priority boost
b) Better accounting for process's run-time
c) Always scheduling the highest priority task
d) Increasing the number of queues

12. In a simple base + bounds approach for virtualization, the address = 0x10, base = 0x200, bound = 0x01. Determine the physical address/validity:

a) 0x201
b) 0x210
c) Invalid/out-of-bounds
d) 0x1F0

13. Determine the total number of addressable location in a 9 bit address:

a) 1024
b) 2048
c) 512
d) 128

14. In address $(0100100101)_2$, the two most significant two bits are used for the segment number. Determine the offset:

a) $(65)_{10}$
b) $(37)_{10}$
c) $(159)_{10}$
d) Need more information to determine offset

15. The Segmentation approach allows sharing of data memory across running processes:

a) True
b) False

Reg No.:_____

Question 2 : (CLO-1, Cognitive level-1 i.e. knowing, PLO-1) **[15]**

1. **[5]** Given the following piece of code, determine the number times the word *test* is printed out. Explain your answer.

```c
#include<stdio.h>

void main(){
  int rc1 = fork();
  int rc2 = fork();
  int rc3 = fork();
  printf("test\n");
}
```

Number of times *test* is printed out: _____

2. **[5]** Write a function to print the *reverse* of a string. The function template is:

```c
void reverse(chr * str)
```

For-example, if the argument is the string "exam", the function should print out "maxe". Use of C Standard Library functions is allowed.

3. **[5]** Write a program that uses the *fork()* system call. The parent should call *fork* once. The child should print out parent's PID and exit. The parent should wait for the child and print "exiting", after the child has exited. Following standard library functions would be useful:

```
pid_t fork(void);
pid_t getpid(void);
pid_t getppid(void);
pid_t wait(int *wstatus);
```

You may answer this question by completing the following code:

```
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>
#include <stdlib.h>

void main(){

  pid_t rc = fork();

  if(                    ){ // execute if fork is unsuccessful
    exit(1);
  }
  else{
    if (                     ){ //only child should execute this

      print("parent's PID %d\n",       ); // fill in the variable name
      exit(1);
    }
    else{                 // only parent should execute this



      print("exiting\n");
      return(1);
    }
  }
}
```

Reg No.:_____

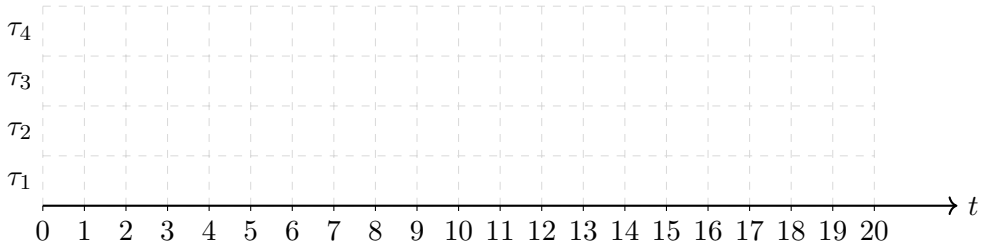
Question 3 : (CLO-2, Cognitive level-2 i.e. comprehention, PLO-2) **[20]**

Task	Computation Time	Period	Deadline
τ_1	1	4	4
τ_2	2	5	5

Table 1: A periodic task-set

1. **[6]** Given four processes in the table below, determine the schedule using shortest-time-to-completion-first algorithm.

Process	Computation Time	Arrival Time
τ_1	7	0
τ_2	2	2
τ_3	4	3
τ_4	3	6

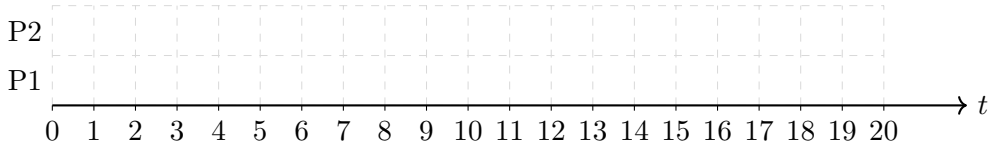


2. **[4]** Determine the turnaround-time and response time for all processes in the above schedule (answer to Question 3.1).

Task	Turnaround Time	Response Time
τ_1		
τ_2		
τ_3		
τ_4		

3. **[6]** Two processes are scheduled using lottery scheduling. The specifics of the processes are given in the table below. Assume that the time slice is one time unit long. Determine and draw the schedule in the figure below. You can assume the average case behavior of the random number generator. Justify your answer.

Process	Computation Time	Arrival Time	Tickets
P1	10	0	80
P2	5	5	20



4. **[9]** Answer the following short questions regarding Multi-Level Feedback Queue (MLFQ):

   (a) **[3]** If there are two processes at in the same queue/priority level, how will they be scheduled?

   (b) **[6]** Priority boost solves two scheduling problems. Explain the two problems, and how they are solved using priority boost.

Reg No.:_____

Question 4 : (CLO-2, Cognitive level-3 i.e. applying, PLO-2) **[20]**

1. **[5]** Assuming the simple base+bound approach, convert virtual address into physical addresses. Following is the configuration:

Address space size = 2k	Physical memory size = 8k
Base register = 0x00001281 (decimal 4737)	Bounds register = decimal 920

   Provide physical address in **hexadecimal format** if the address is within bounds. If the address is out of bounds, write down "SEGMENTATION FAULT".

   - VA 0: 0x0000020b (decimal: 0523) : _____
   - VA 1: 0x0000060c (decimal: 1548) : _____
   - VA 2: 0x00000301 (decimal: 0769) : _____
   - VA 3: 0x00000669 (decimal: 1641) : _____
   - VA 4: 0x000002b6 (decimal: 0694) : _____

2. **[10]** Assuming the segmentation approach, convert virtual addresses into physical addresses. For each virtual address, either write down the physical address it translates to in **hexadecimal** OR write down that it is an out-of-bounds address (a "SEGMENTATION FAULT"). For this problem, you should assume a simple address space with two segments: the top bit of the virtual address can thus be used to check whether the virtual address is in segment 0 (topbit=0) or segment 1 (topbit=1). Note that the base/limit pairs given to you grow in different directions, depending on the segment, i.e., **segment 0 grows in the positive direction**, whereas **segment 1 in the negative**. Following is the configuration:

Address space size = 4k	Physical memory size = 64k
Segment 0 base: 0x00001961 (decimal 6497)	Segment 0 bound : decimal 1820
Segment 1 base: 0x00006109 (decimal 24841)	Segment 1 bound : decimal 1709

   - VA 0: 0x0000077c (decimal: 1916) : _____
   - VA 1: 0x0000088e (decimal: 2190) : _____
   - VA 2: 0x00000fa7 (decimal: 4007) : _____
   - VA 3: 0x00000215 (decimal: 0533) : _____
   - VA 4: 0x00000abd (decimal: 2749) : _____

Reg No.:_____

3. **[5]** Assuming the following address configuration answer the questions stated below.

   Total number of bits for address = 16 Number of bits used for segment number = 2

   Determine:

   - **[2]** Maximum addressable memory locations : _____
   - **[1]** Maximum number of segments : _____
   - **[2]** Maximum addressable locations within each segment : _____