CS 3113 Introduction to Operating Systems
Final Exam
December 13, 2019

General instructions:

- Please wait to open this exam booklet until you are told to do so.
- This examination booklet has 13 pages. You also have been issued a bubble sheet.
- Write your name, university ID number and date, and sign your name below. Also, write your name and ID number on your bubble sheet, and fill in the bubbles for your ID.
- The exam is closed book, notes and electronic devices. The exception is that you may have one page of personal notes (double sided).
- The exam is worth a total of 137 points (and 15% of your final grade).
- You have 2 hours to complete the exam. Be a smart test taker: if you get stuck on one problem go on to the next.
- Use your bubble sheet to answer all multiple-choice questions. Make sure that the question number and the bubble row number match when you are answering each question. If you cannot effectively erase an incorrect answer, mark an 'X' over it.

On my honor, I affirm that I have neither given nor received inappropriate aid in the completion of this exam.

Signature: _____

Name: _____

ID Number: _____

Date: _____

Question	Points	Score
Files and File Systems	24	
Process Scheduling	40	
System Calls and Protection	9	
Processes and Threads	22	
Characters and Strings	8	
Resource Sharing	34	
Total:	137	

Part I. Files and File Systems

1. (2 points) True or False? When a process opens a file and then forks a child process, the two processes share the file offset.

   A. True B. False

2. (4 points) Assume that a file of length 250 has been opened for reading and that the current offset is 223. What is the return value of the following read() call:

```
int ret = read(fd, buf, 37);
```

   A. 27 B. 37 C. 223 D. 250 E. Answer not shown

3. (4 points) Assume that a disk block stores I inodes and that there are multiple, sequential blocks that store inodes. Assume also that the first inode block is disk block K. Given that we would like to access inode j, which block and element of the block, respectively, must we access?

   A. $I + j/K$ and $j\%K$
   B. $K + j/I$ **and** $j\%I$
   C. $I + j\%K$ and j/K
   D. $K + j\%I$ and j/I
   E. Answer not shown

4. (4 points) Assume that a directory is deleted from the OUFS; the directory's inode index is 18. Before the deletion, the first three entries of the inode allocation table are: $0xB4$, $0x3E$ and $0xAC$. After a successful deletion, what is the new state of the inode allocation table (first 3 entries)?

   A. $0xB0$, $0x3E$ and $0xAC$
   B. $0xB4$, $0x1E$ and $0xAC$
   C. $0xB4$**,** $0x3E$ **and** $0x8C$
   D. $0xB4$, $0x3E$ and $0xA8$
   E. Answer not shown

5. (2 points) True or False? It is faster to sequentially access blocks on a hard disk that are contiguous than if they were randomly distributed on the disk.

   A. True B. False

6. (4 points) Assume the initial contents of file *myfile2* and the execution of the following block of code. What are the final contents of myfile2? Assume that there are no errors.

myfile2 initial contents:

```
able-was-I-ere-I-saw-elba
```

Code block:

```
char buf[100];

int fd = open("myfile2", O_RDWR);

lseek(fd, 11, SEEK_SET);

read(fd, buf, 4);

buf[3] = 0;

printf("%s\n", buf);

close(fd);
```

A. -ere **B. ere** C. ere- D. re- E. Answer not shown

7. (4 points) Assume the initial contents of file *myfile* and the execution of the following block of code. What are the final contents of myfile? Assume that there are no errors with permissions and that there are no newlines.

myfile initial contents:

```
pass
```

Code block:

```
FILE *fp = fopen("myfile", "a");

if(fp == NULL){
  printf("Error");
  exit(-1);
}

fprintf(stdout, "_the exam");

fclose(fp);
```

A. pass B. _the exam C. pass_the exam D. There is an error
E. Answer not shown

Part II. Process Scheduling

8. (4 points) Consider the following processes, and arrival and burst times:

Process	**Arrival Time**	**Burst Length**
P1	0	8
P2	1	2
P3	2	6

According to the **shortest-time-remaining-first** algorithm, what is the average wait time for these processes?

A. 3 B. 4 C. 5 D. 6 E. Answer not shown

9. (4 points) Assuming that the same set of processes are scheduled according to the **shortest-job-first** algorithm (i.e., non-preemptive), what is the average wait time?

A. 3 B. 4 **C. 5** D. 6 E. Answer not shown

10. (4 points) Assuming that the same set of processes are scheduled according to the **round robin** algorithm with a quantum of 5, what is the average wait time? Assume that newly arriving jobs are placed at the end of the round robin queue.

A. 3 B. 4 C. 5 D. 6 **E. Answer not shown**

11. (4 points) Consider a file system in which inodes contain a block index table, with one such table configured accordingly:

Index	Block
0	17
1	23
2	18
3	21

Assume a block size of 16. Which physical block is logical byte 45 of the file stored in?

A. 17 **B. 18** C. 21 D. 23 E. Answer not shown

12. (4 points) Which byte within the physical block is byte 45 stored in?

A. 2 B. 3 C. 5 **D. 13** E. Answer not shown

13. (4 points) Consider the following directory listing under a contiguous block allocation scheme with a block size of 8:

File	Start Block	N Blocks
index.html	23	2
project0.html	53	4
project1.html	41	8
project2.html	8	13

Which physical block is logical byte 27 of project1.html stored in?

A. 41 B. 42 C. 43 **D. 44** E. Answer not shown

14. (4 points) Which byte within the physical block is byte 27 of project1.html stored in?

A. 2 **B. 3** C. 5 D. 7 E. Answer not shown

15. (4 points) What is the logical byte that corresponds to physical block 11, byte 7?

A. 23 B. 25 C. 27 **D. 31** E. Answer not shown

16. (4 points) Consider the following processes and burst times:

Process	**Burst Length**
P1	7
P2	4
P3	10

Assuming that all of the processes arrive at time zero, but in the following order: P1, P2, P3, what is the average wait time if the processes are scheduled according to the **first-come-first-served** algorithm?

A. 5 **B. 6** C. 7 D. 13 E. Answer not shown

17. (4 points) Assuming that the same set of processes are scheduled according to the **shortest-job-first** algorithm, what is the average wait time?

A. 5 B. 6 C. 7 D. 13 E. Answer not shown

Part III. System Calls and Protection

18. (3 points) Which of the following is **not** a class of system call.
    A. Process control
    B. Memory management
    C. Communication
    D. Device management
    E. All are classes of system calls

19. (2 points) True or False? *strcmp()* is a system call.
    A. True **B. False**

20. (2 points) True or False? A system call allows the kernel to access user space code.
    A. True **B. False**

21. (2 points) True or False? The file system is always manipulated by code executing in a kernel mode.
    A. True **B. False**

Part IV. Processes and Threads

22. (2 points) True or False? Creating a new thread is more expensive than creating a new process.

A. True **B. False**

23. (3 points) In a thread library, what is the complement to *fork()*?

A. exit() **B. join()** C. return() D. spoon() E. Answer not shown

24. (3 points) In the many-user-threads to one-kernel-thread approach to thread management, scheduling of thread execution is typically done in:

A. Kernel mode **B. User mode** C. Answer not shown

25. (2 points) True or False? Different threads of the same program have their own stacks.

A. True B. False

26. (4 points) What number is printed by this program?

```
int main(int argc, char** argv)
{
  int p1[2];

  pipe(p1);

  int pid;
  int val = 42;

  if((pid = fork()) == -1) {
    fprintf(stderr, "Error forking\n");
    exit(-1);
  }else if(pid > 0){
    close(p1[0]);

    val = 7;
    write(p1[1], &val, sizeof(int));

    wait(NULL);

  }else{
    close(p1[1]);

    int val2 = 3;
    read(p1[0], &val2, sizeof(int));
    val2 += val;

    printf("%d\n", val2);
  }
}
```

A. 10 B. 45 **C. 49** D. Answer not shown

27. (4 points) How many stars are output by this program?

```
int main(int argc, char** argv)
{
  for(int i = 0; i < 2; ++i) {
    int pid = fork();
    write(1, "*", 1);

    if(pid > 0) {
      wait(NULL);
    }
  }
}
```

A. 3 B. 4 **C. 6** D. 8 E. Answer not shown

28. (4 points) Suppose that 75% of a computational task can be parallelized. According to Amdhal's law, with 100 processors, what is the effective speed-up?

A. About 1.5 **B. About 4** C. About 25 D. About 100
E. Answer not shown

Part V. Characters and Strings

29. (4 points) What is printed by the following block of code? (do not worry about spaces and newlines in your answer)

```
char str[100];

strcpy(str, "IBM");
strcpy(&str[2], "HAL");

printf("%s\n", str);
```

A. HAL B. IBM **C. IBHAL** D. IBMHAL E. Answer not shown

30. (4 points) What is output by the following block of code? (do not worry about spaces and newlines in your answer)

```
char str[100];

strcpy(str, "Out: 1,5,27");

int len = strlen(str);

for(int i = 0; i < len; ++i) {
  if(str[i] >= '0' && str[i] <= '9')
    str[i] = (str[i] - '0' + 5)%10 + '0';
}

printf("%s\n", str);
```

A. Out: 1,5,2
B. Out: 1,5,27
C. Out: 6,0,2
D. Out: 6,0,72
E. Answer not shown

Part VI. Resource Sharing

31. (4 points) True or False? According to the Banker's algorithm, the following system state is safe.

Claim/Max Matrix

	Camera	**Printer**	**Bluetooth**
Process A	2	1	2
Process B	5	5	3
Process C	5	4	4
Process D	1	0	2

Current Allocation

	Camera	**Printer**	**Bluetooth**
Process A	1	1	0
Process B	2	4	1
Process C	4	2	3
Process D	1	0	1

Total Resources

Camera	**Printer**	**Bluetooth**
8	7	6

A. True **B. False**

32. (4 points) Consider the following solution to the dining philosophers problem where the philosophers are: $P_1, P_2, ..., P_K$, and the chopsticks are $C_0, C_1, ...$ (each is a binary semaphore initialized to state '1').

Each philosopher i uses the following algorithm:

```
while(true){
    think()
    wait(chopstick[i])
    wait(chopstick[i/2])
    eat()
    signal(chopstick[i/2])
    signal(chopstick[i])
}
```

Can deadlock happen?

A. Yes **B. No**

33. (3 points) Which condition of deadlock is prevented by requiring that all resource requests by a process are made simultaneously?

A. Hold and wait B. No preemption C. Exclusion D. Circular wait

34. (4 points) Assume that there is exactly one unit of each resource, and that the allocations and requests are as follows:

- P1: allocated R1 and requested R2
- P2: requested R1 and R2

In this state, is the resource manager allowed to allocate resource R2 to either P1 or P2?

A. P1: No; P2: No
B. P1: No; P2: Yes
C. P1: Yes; P2: No
D. P1: Yes; P2: Yes (either P1 or P2, not both at the same time)

35. (4 points) True or False? According to the Banker's algorithm, the following system state is safe.

Claim/Max Matrix

	Camera	**Printer**	**Bluetooth**
Process A	7	5	4
Process B	7	2	3
Process C	3	2	5
Process D	4	0	2

Need Matrix

	Camera	**Printer**	**Bluetooth**
Process A	4	3	3
Process B	4	0	1
Process C	2	1	4
Process D	1	0	1

Total Resources

Camera	**Printer**	**Bluetooth**
11	5	6

A. True B. False

36. (3 points) Which of the following is a hardware solution to the mutual exclusion problem?

A. Binary semaphore
B. Counting semaphore
C. Mutex
D. Test-and-set
E. Answer not shown

37. (4 points) Below is an implementation of a barber who can perform hair cuts on people, one at a time. Furthermore, there is a function that any number of people can call at any time (including at the same time) to get a hair cut. **customer** and **barber_done** are counting semaphores.

```
barber()
{
  while(true){
    wait(customer);
    cut_hair();
    signal(barber_done);
  }
}

customer_get_hair_cut()
{
  signal(customer);
  wait(barber_done);
  pay();
}
```

How should the two semaphores be initialized before either of these functions is called? The barber cannot cut hair unless there is a customer and a customer cannot pay until the work is done.

A. $customer = 0; barber\_done = 0$
B. $customer = 0; barber\_done = 1$
C. $customer = 1; barber\_done = 0$
D. $customer = 1; barber\_done = 1$
E. Answer not shown

38. (2 points) True or False? If the Banker's algorithm determines that a state is **unsafe**, then **deadlock** is guaranteed to happen.

A. True **B. False**

39. (2 points) True or False? **Deadlock** is a special case of **starvation.**

A. True **B. False**

40. (4 points) Assume that there is exactly one unit of each resource, and that the allocations and requests are as follows:

- P1: allocated R1 and R2
- P2: requested R1 and R3
- P3: requested R2 and allocated R3

Has deadlock occurred?

A. Yes **B. No**