CS 3113 Introduction to Operating Systems
Midterm
October 8, 2019

General instructions:

- Please wait to open this exam booklet until you are told to do so.
- This examination booklet has 10 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 1.25 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	25	
Memory and Memory Management	30	
Program Compilation	13	
System Calls and Kernels	22	
Processes	29	
Characters and Strings	18	
Total:	137	

Part I. Files and File Systems

1. (6 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:

```
Tweedle
```

   Code block:

```
int fd = open("myfile", O_WRONLY | O_APPEND | O_TRUNC);

if(fd < 0){
  printf("Error");
  exit(-1);
}

write(fd, "dee", 3);

close(fd);
```

   A. dee B. Tweedle C. Tweedledee D. There is an error
   E. Answer not shown

2. (3 points) True or False? When two processes simultaneously and independently open the same file name, they share file offsets.

   A. True B. False

3. (3 points) True or False? In a file system with an *acyclic graph directory* structure, a single file can have multiple paths that reference it.

   A. True B. False

4. (6 points) Assume the initial contents of file *myfile2* and the execution of the following block of code. What is printed by the program?

   myfile2 initial contents:

```
foo#bar#baz
```

   Code block:

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

if(fd <= 0){
  printf("Error");
  exit(-1);
}

if(lseek(fd, 5, SEEK_SET) < 0) {
  printf("Error");
  exit(-1);
}

char buffer[100];

if(read(fd, buffer, 3) != 3) {
  printf("Error");
  exit(-1);
}

buffer[3] = 0;
printf("%s\n", buffer);

close(fd);
```

   A. foo   B. bar   C. ar#   D. Error   E. Answer not shown

5. (3 points) True or False? A file descriptor that references a file includes a buffer to temporarily store outgoing/incoming bytes.

   A. True   B. False

6. (4 points) Which of the following **is not** a category of control within a Unix *access control list*?

   A. Admin   B. Group   C. Other   D. User/Owner   E. Answer not shown

Part II. Memory and Memory Management

7. (5 points) When an interrupt occurs, what information **is not stored** on the stack?

   A. A copy of the registers that will be used by the interrupt handler
   B. Local variables for the interrupt handler
   C. Buffered values for the data received from the I/O device
   D. Address of the interrupted instruction
   E. None of the above; the interrupt handler does not use the stack

8. (3 points) True or False? In the typical process (default state when first created), memory accessible by a process is exclusively owned by that process.

   A. True B. False

9. (5 points) Below is a list of properties of memory (assuming a C compiler). Select the one item that is **not true** for the stack.

   A. Fast allocation
   B. Size determined at compile time
   C. Local variable storage
   D. Persistent after function returns
   E. All are true

10. (6 points) What is the value of **c** after the following code is executed?

```
unsigned char a = 0xA3;
unsigned char b = a << 2;
unsigned char c = b >> 4;
```

   A. 0x8 B. 0x28 C. 0x51 D. 0xA5 E. Answer not shown

11. (5 points) As you travel down the memory hierarchy (from the smallest-sized memory) which of the following is generally true?

   A. Decreasing cost (in dollars) per bit and increasing access time
   B. Increasing cost per bit and increasing access time
   C. Increasing cost per bit and decreasing access time
   D. Decreasing cost per bit and decreasing access time
   E. Answer not shown

12. (6 points) What is the value of **c** after the following code is executed?

```
unsigned char a = 0xA3;
unsigned char b = 0x31;
unsigned char c = a | b;
```

   A. 0x0 B. 0x1 C. 0x21 D. 0xB3 E. Answer not shown

Part III. Program Compilation

13. (4 points) The *gcc* executable performs a number of different functions. Which one of these is responsible for creating object files?

    A. Linker   B. C Preprocessor   C. Compiler   D. Answer not shown

14. (6 points) Consider a program that is implemented as two distinct source files (part1.c and part2.c) to create a single executable, *whole*. Which one line can be changed to fix the bug that prevents the executable from being created when we type *make all* into the shell?

```
 1  CC = gcc
 2  CFLAGS = -c -Wall -g
 3  EXEC = whole
 4  PARTS = part1.o
 5  all: $(EXEC)
 6
 7  part1.o: part1.c
 8          $(CC) $(CFLAGS) part1.c -o part1.o
 9
10  part2.o: part2.c
11          $(CC) $(CFLAGS) part2.c -o part2.o
12
13  $(EXEC): $(PARTS)
14          $(CC) $(CFLAGS) $(PARTS) -o $(EXEC)
15
16  clean:
17          rm $(EXEC)
```

    A. 4   B. 10   C. 11   D. 13   E. 14

15. (3 points) True or False? Assuming no bugs in the compilation/linking part of the above Makefile, the *clean* command deletes all files that *make all* creates.

    A. True   B. False

Part IV. System Calls and Kernels

16. (3 points) True or False? In Linux, the *bash* shell program executes in kernel space.
    A. True B. False

17. (4 points) Which of the following **is not** a Unix system call?
    A. fork() B. malloc() C. open() D. read() E. sleep()

18. (4 points) Which type of kernel attempts to minimize the amount of OS functionality that must be executed in kernel mode?
    A. Hybrid kernel B. Layered kernel C. Microkernel D. Modular kernel
    E. Monolithic kernel

19. (5 points) Which of the following **is not** a function of a typical kernel?
    A. Creating new processes
    B. Moving data into and out of I/O devices
    C. Scheduling processes for execution on the CPU
    D. Management of the memory allocation to processes
    E. Creating stack frames during function calls

20. (3 points) True or False? In Unix, communication through **ordinary pipes** is mediated by the kernel.
    A. True B. False

21. (3 points) True or False? A system call can only be called in kernel mode.
    A. True B. False

Part V. Processes

22. (5 points) What information is kept in the Process Control Block? (select one)
    A. The name of the executable
    B. A copy of the register values
    C. The memory location of all functions in a program
    D. The location of the executable on the disk
    E. The offset of a file that was opened by the process

23. (4 points) In the 5-state process model, a process in the **blocked or waiting state** can move to which of the following states (assuming the conditions are right)?
    A. New B. Ready C. Running D. Terminated E. Answer not shown

24. (6 points) What is output by this program? Assume no errors in executing system calls.

```
int main(int argc, char** argv)
{

  for(int i = 1; i < 6; ++i) {
    if(fork() == 0) {
      ++i;
      printf("%d", i);
      fflush(stdout);
    }else{
      wait(NULL);
      return(0);
    }
  }

  return(0);
}
```

A. 135 B. 246 C. 12345 D. 23456 E. Answer not shown

25. (3 points) True or False? In Linux, all processes have a parent process.
    A. True B. False

26. (6 points) What is the output of the following program? Ignore newlines in the answers.

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

  if(pipe(fds) == -1) {
    fprintf(stderr, "Error opening pipe\n");
    exit(-1);
  }

  int pid;

  if((pid = fork()) == -1) {
    fprintf(stderr, "Error forking\n");
    exit(-1);
  }else if(pid > 0){
    close(fds[1]);
    int val = 42;
    read(fds[0], &val, sizeof(int));
    printf("X: %d\n", val);
    wait(NULL);
  }else{
    close(fds[0]);
    int val = 24;
    if(write(fds[1], &val, sizeof(int)) != sizeof(int)){
      fprintf(stderr, "Error reading int\n");
      exit(-1);
    }
    sleep(1);
    printf("Y: %d\n", val);
  }
}
```

A. X: 24 Y: 24 B. X: 24 Y: 42 C. X: 42 Y: 24 D. X: 42 Y: 42
E. Answer not shown

27. (5 points) A **context switch** refers to which of the following?

A. A program calling execlp() to switch to executing a completely different program within the same process
B. Starting to execute an interrupt service routine in the middle of executing a user space program
C. Transitioning from user mode to kernel mode (or vice versa)
D. Moving one process off the CPU and another process into its place
E. Answer not shown

Part VI. Characters and Strings

28. (6 points) What is output by this program?

```
int main(int argc, char **argv)
{
  char buf1[100];
  strcpy(buf1, "Boa tarde");
  buf1[4] = 0;
  strcat(buf1, "noite");

  printf("%s\n", buf1);
}
```

A. Boa B. Boa noite C. Boa tarde D. Boa tardenoite
E. Answer not shown

29. (6 points) What is output by this program when executed as indicated?

```
int main(int argc, char **argv)
{
  if(*argv[1] > *argv[2])
    printf("rabbit\n");
  else if(*argv[1] < *argv[2])
    printf("cat\n");
  else
    printf("hatter\n");
}
```

```
./a.out slythy toves
```

A. rabbit B. cat C. hatter D. nothing is printed E. Answer not shown

30. (6 points) Consider the following program:

```
const int LEN = 100;

int main(int argc, char **argv)
{
  int buf[LEN];

  for(int i = 0; i < LEN; ++i) {
    buf[i] = 0;
  }

  char input[LEN];

  if(fgets(input, LEN, stdin) != NULL) {
    for(int i = 0; i < strlen(input); ++i) {
      if(input[i] >= 'a')
         buf[input[i] - 'a']++;
    }
  }

  printf("%d\n", buf[5]);
}
```

What output is generated when the following string is typed?

```
foo bar baz foobar foobaz barbaz foobarbaz
```

A. 0 B. 3 C. 4 D. 8 E. Answer not shown