CS 376	**Operating Systems**	Test 1
Name:	Fall 1997	Max Pts: 32

Important: This is an open book test. You can use any books, notes, or paper. *Do not log into the computer during the test.* Any calculations and rough work can be done on the back side of the test pages. You will loose five points for not writing your name.

1. [2pt] Write an expression to add two variables in C shell.

2. [5 pt] In Bourne shell, you have to write a script that needs to use a temporary file. A well-accepted way to create such files is to call it `tmp.xxxxx` where `xxxxx` is the process id of the shell. How can you write the output of a command `foo` to this file. Moreover, this file must be deleted when the user issues a `^c` command. How will you achieve that in Bourne shell (Give complete syntax of the command)?

3. [5 pt] I want my system to be backed up automatically at 3:00 am every Saturday. I have written a script to do the same, and called it `/usr/local/bin/backup`. I certainly do not want to get up in the middle of the night to start the backup. What commands can I issue to achieve this without me having to disturb my sleep? Assume that I have already taken care of hardware problems like inserting tape in the tape drive.

4. [5 pt] Describe the main differences between symmetric and asymmetric multiprocessing (MIMD).

5. [5 pt] Look at the following implementation of bakery algorithm. Is there a problem in here? Which of the three requirements of critical-section problem does it violate?

```
while ( 1 )
{
    choosing[i] = 1;    /* True  */
    number[i] = 1 + max(number, n);
    choosing[i] = 0;    /* False */
    for ( j = 0; j < n; j++ )
    {
       while ( choosing[j] );
       while ( (number[j],j) < (number[i],i) && number[j] );
    }
    critical_section();
    number[i] = 0;
    remainder_section();
}
```

6. [5pt] Consider the following sequence of commands ($ is the shell prompt):

```
$ grep main a.c b.c c.c > grepout &
$ wc -l < grepout &
$ rm grepout &
```

The ampersand at the end of each command line informs the shell to run the command in the background, and it can execute each command line in parallel. Why is this not equivalent to the following command line?

```
$ grep main a.c b.c c.c | wc -l
```

7. [5 pt] The following code is for process p_0. Similar code exists for process p_1. Does it satisfy the requirements for critical section problem?

```
extern int flag[2];          /* Shared Boolean variable              */
extern int turn;             /* Shared variable to indicate turn     */

main()                       /* Code for process 0                   */
{
    while ( 1 )
    {
        /* Entry to critical section */

        flag[0] = 1;
        while ( flag[1] )
            if ( turn == 1 )
            {
                flag[0] = 0;
                while ( turn == 1 );
                flag[0] = 1;
            }

        /* Enter critical section                                    */

        critical_section();

        /* Exit from critical section                                */

        turn = 1;
        flag[0] = 0;

        /* Remainder section                                         */

        remainder_section();
    }
}
```