All questions are equally weighted.

1. (a) Where is the free space-list (free data blocks) saved?

   (b) What function is used to make sure this data structure is correctly updated before events such as the system going down?

   (c) What place(s) is(are) checked for permissions data when "/a/b/c" is opened for reading?

2. AR implementation

   (a) What is the function of the magic string for ar archives?

   (b) What is the major difference in the structure of archives that contain binary files from those that don't?

   (c) Assuming f1 is a new file, in performing "ar -r arch f1", how does ar get the owner of the file "f1" and what steps does it have to do before writing the header?

   (d) Supposing that one header had just been read, give code showing how move to the next "file header" in the archive file.

3. CPP

   (a) define a macro MAX3(a,b,c) that returns the maximum of three numbers. If you like you can use MAX(a,b) that returns the maximum of 2 numbers.

   (b) How many cpp transformations (macro expansions) are performed on the line
   while ((c = getc(stdin)) != -1)
   ?

   (c) What is the result of the first transformation?

   (d) What is the result of the next transformation (if there is a next one)?

4. Memory Layout

   (a) Show the memory layout of a UNIX process.

   (b) Give a section of code that will get the SHELL variable from the environment.

   (c) What happens to the memory layout when you add a couple of items to the environment? (i.e., what moves where?)

5. Given that the program shown below

```
#include <stdio.h>
int *p = NULL;
void a (int );

main(){
   int n;
   n = 4;
   a(n);
}

void a (int i)
{
    int j = i;
    if(j > 0){
       if (p == NULL) p = &j;
       else{
           printf("the size is %d\n", p-&j);
           p = &j;
       }
       a(j-1);
    }
}

produces the output
the size is 30
the size is 30
the size is 30

What is 30 the size of?


What elements (data items) are in that area?
```

6. Assume the UFS format for this question.

   (a) Show a diagram and explain how adding one byte to a file might add three data blocks to the file's space requirements.

   (b) Assuming that blocks are 8K and that each block can hold 2048 pointers. At which size would you expect for adding one character to a file add three blocks. Leave you answer as an expression, such as 45*8K + 8K*8K + 47.

7. Write a complete program that when run will create another process and the child process print out a message containing its parent's process ID and the parent process should print out the child's pid.

8. GDB

(a) How do you compile so that you can use gdb to debug your programs?

(b) How do you set a breakpoint on line 35?

(c) How do you single step (stepping into functions)?

(d) How do you show the stack of activation records that is active?

9. Write a C program "newerthan t f1 f2 ... fn", that takes an integer t(=time) in days and a list of files f1 f2 ... fn and prints the name of those files that were created since t days ago.

10. TAKE-HOME due Tuesday Oct 2 Midnight
Make a directory named Test1 and include in it a Makefile, name the executable "test1" and have clean and test targets, also.

    (a) **Undergraduate** Write a program that will take either one or no arguments. If there is an argument it should be a path to a directory otherwise the current directory (".")is used. This program should find the total number of blocks used by all of the files in this directory. You may implement this so that it does not recursively enter subdirectories. *For extra credit you can recursively count subdirectories.*

    (b) **Graduate** Write a function getelement() that allocates space for an "element", reads and intializes the element from the stdin and returns a pointer to the element. An element is a structure containing an integer "val" and "next," a pointer to the next element in the list.

    Also assume that a global pointer "head" points to a list of elements that are to be re-processed much as in the manner that ungetch-getch (or ungetc-fgetc) reprocess characters. Write a function nextelem that checks head and if it is not null returns a pointer to the first element of the list. If head is null then getelement is called and the pointer returned from the call is the value returned as the value of nextelem.

    Write a test program that tests this function.

Send questions on the Take-Home via email. I will be in some on Saturday and/or Sunday. **Read email! because I respond to everyone.**