CS 61CL (Clancy/Culler) **Exam 1**
Fall 2008

Read and fill in this page now.
Do NOT turn the page until you are told to do so.

Your name: _____

Your login name: _____

Your lab section day and time: _____

Your lab t.a.: _____

Problem 0	_____		Total:		_____	/30
Problem 1	_____	_____				
	_____	_____	Problem 4	_____	_____	
Problem 2	_____			_____	_____	
Problem 3	_____	_____	Problem 5	_____	_____	

This is an open-book test. You have approximately two hours to complete it. You may consult any books, notes, or other paper-based inanimate objects available to you. You may not use any electronic devices.

To avoid confusion, read the problems carefully. If you find it hard to understand a problem, ask us to explain it. If you have a question during the test, please come to the front or the side of the room to ask it. The more difficult problems are at the end of the exam.

This exam comprises 15% of the points on which your final grade will be based. Partial credit may be given for wrong answers. Your exam should contain six problems (numbered 0 through 5) on nine pages. Please write your answers in the spaces provided in the test; in particular, we will not grade anything on the back of an exam page unless we are clearly told on the front of the page to look there.

Some students are taking this exam late. Please do not talk to them, mail them information, or post anything about the exam to news groups or discussion forums until after Thursday.

Relax—this exam is not worth having heart failure about.

Problem 0 (2 points)

Put your login name on each page. Also make sure you have provided the information requested on the first page.

Problem 1 (4 points)

Show your work on each part of this problem for full credit.

Part a

Give the 8-bit base 2 unsigned representation of the decimal value 244.

Part b

Interpret your answer to part b as an 8-bit two's complement *signed* decimal value.

Part c

Give the base 16 representation of the decimal value 244.

Part d

Give the base 3 representation of the decimal value 244.

Problem 2 (4 points)

The C string library contains a function named strchr that, given arguments char *s and char c, returns a pointer to the first occurrence of c in the string pointed to by s. If c does not occur in the string, strchr returns NULL. The terminating null character is considered part of the string; therefore if c is '\0', strchr locates the terminating '\0'.

Implement the strchr function.

```
char *strchr (char *s, char c) {
```

Problem 3 (6 points)

Each part of this problem involves completing the replace function described below. Given an array of strings named table as argument along with an index k into the table and a new string s, replace essentially performs the assignment

```
table[k] = a copy of s;
```

It need not free the replaced storage. The replace function might be called as follows.

```
char* table [ ] = {"mike", "clancy"};
replace (table, 1, "hello");
/* table now contains the strings "mike" and "hello".
```

Part a

Complete the replace function, using array notation, i.e. square brackets, where possible. Your solution need not use all the lines.

```
void replace ( ____________________________ ) {

   ____________________________________________________________ ;

   ____________________________________________________________ ;
   /* includes a call to malloc */

   ____________________________________________________________ ;
   /* includes a call to strcpy */

   ____________________________________________________________ ;
}
```

Problem 3, continued

Part b

Complete the replace function, *without using* any array notation (square brackets). Again, your solution need not use all the lines.

```
void replace ( ____________________________ ) {

   ____________________________________________________________ ;

   ____________________________________________________________ ;
   /* includes a call to malloc */

   ____________________________________________________________ ;
   /* includes a call to strcpy */

   ____________________________________________________________ ;
}
```

Problem 4 (7 points)

Part a

Provide the type declaration for a struct node that contains an int named val and pointer named ptr to a value of type struct node.

Part b

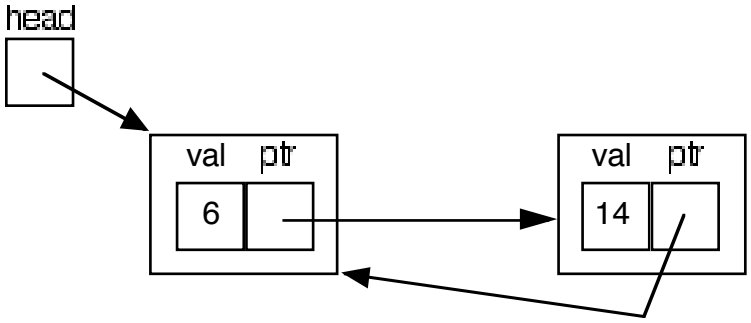
Write a function min that takes as argument a pointer to a list of nodes and returns the minimum value in the list. Assume the argument list is not null and noncircular, and that the last node in the list is indicated by a null ptr pointer.

```
int min (struct node *p) {
```

Problem 4, continued

Part c

Using your declaration from part a, provide a program segment that dynamically allocates the structure represented in the diagram below.



Part d

Explain in a sentence or two what would happen if head in the structure in part c was passed as argument to the function you wrote for part b.

Problem 5 (7 points)

Both parts of this problem involve the following code.

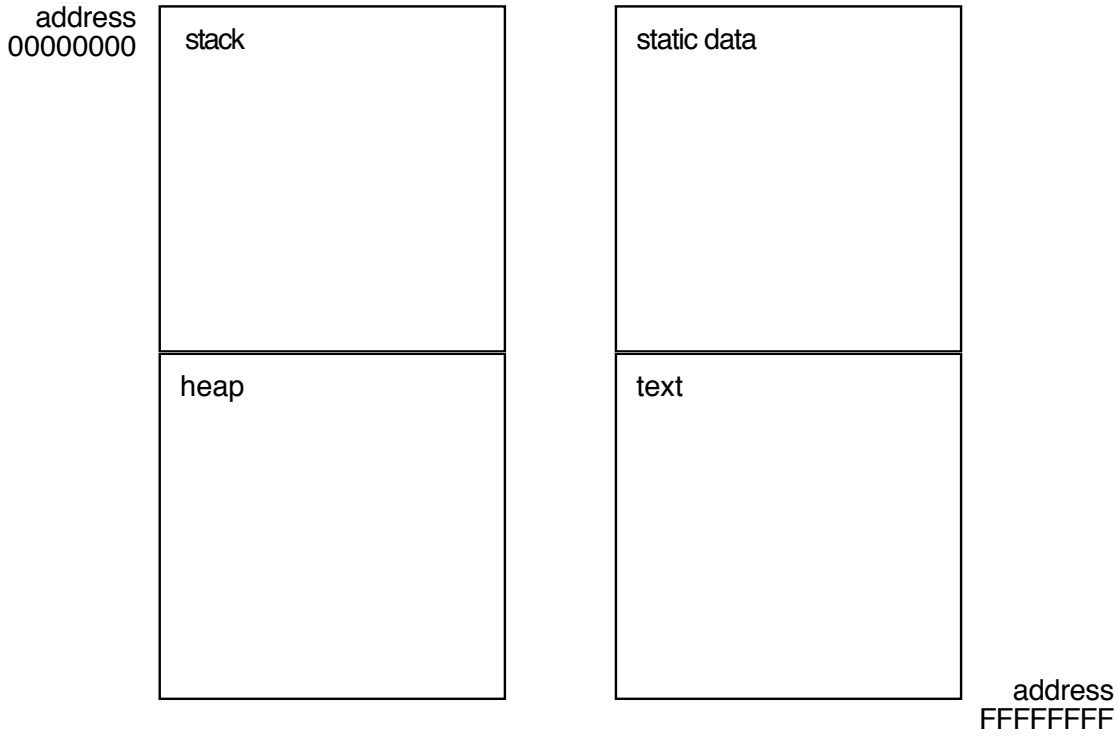
```
char facName [ ] = "Mike";

char *response (char *name) {
   int k, helloLen;
   char *p;
   helloLen = strlen ("Hello, ");
   p = (char *) malloc (sizeof(char) * (helloLen + strlen (name)));
   p = "Hello, ";
   for (k=0; k<=strlen (name); k++) {
      p[k+helloLen] = name[k];
   }
}

int main ( ) {
   printf ("%s\n", response (facName));
   return 0;
}
```

Part a

Given below is a diagram of memory. Draw labeled boxes on the diagram that represent the location and contents of the variables facName, name, and p. Your answer for p should reflect its state immediately after the call to malloc. Draw a pointer value as an arrow. If the contents of a variable cannot be determined, put a question mark in the corresponding box.



Problem 5, continued

Part b

The response function (reprinted below for your convenience) has at least three errors. Find and fix them, and explain why your fixes correct the bugs.

```
char *response (char *name) {
   int k, helloLen;
   char *p;
   helloLen = strlen ("Hello, ");
   p = (char *) malloc (sizeof(char) * (helloLen + strlen (name)));
   p = "Hello, ";
   for (k=0; k<=strlen (name); k++) {
      p[k+helloLen] = name[k];
   }
}
```