CS 402: Systems Programming

Midterm Examination

This test has 7 problems worth a total value of 100 points. Note that some problems are worth more than others. Be sure to read over all of the problems carefully before answering them. A little thought will help you avoid making careless errors.

You may use your class texts, notes, handouts, class newsgroups postings and listings of your assignments. You may not use any other kind of reference materials on this examination.

1. (15 points) Beck shows how the SIC/XE architecture can do indirect memory addressing. Does MIPS support indirect memory addressing? If so, give an example MIPS instruction which does so, and explain its use. If not, give a sequence of MIPS instructions which performs an indirect read from memory.

2. (15 points) Show the output of the following program fragment:

```c
char p1[100], *p2, *p3;
p3 = p1;
p2 = (char *)malloc(100*sizeof(char));
strcpy(p2, "Welcome. We've been waiting for you.");
while (*p3++ = *p2++);
p1[10] = '\0';
printf("%s\n", p1);
```

3. (5 points) Does the following program fragment assign the value 2 to the variable p? Give an explanation of no more than two sentences.

```c
int p, x = 7, y = 0;
p = (x=y) ? 7 : x + 2;
```

4. (20 points) Write a fcn with the following prototype:

```c
void *odd_strcpy(char *, char *);
```

where the odd characters (i.e. characters with indices 1, 3, 5 ...) of the second string are copied to the first string. For example, the string "abcdefg" would be copied as "bdf" by this fcn. If the second string is of length 0 or 1, a string of length zero is returned. Assume that storage has already been allocated for the string that the copy will be made to. You do not need to worry about space limitations in the string.

5. (5 points) What field in the opcode table for SIC/XE is unnecessary for the opcode table for MIPS? Why is this?

6. (X20 points) What memory and register value changes will the following MIPS machine code sequence cause?

```
3C03AAAA
2463AAAA
30638888
```
7. (20 points) Using dynamic allocation, the following struct, and the variable root:

```c
struct node {
    int this_val;
    struct node *left, *right;
} *root;
```

show the C code fragment necessary to construct a data structure corresponding to the following figure:

![Binary Tree Diagram]

where the value in the nodes is this_val.