

Name \_\_\_\_\_

Section \_\_\_\_\_

**CS2303 C08**  
**Systems Programming Concepts**  
**Practice Mid Term Exam**  
**January 30, 2008**

Question	Points	Score
0	1	
1	5	
2	4	
3	5	
4	20	
5	12	
6	16	
7	18	
<b>Total</b>	<b>80</b>	

Trivia Question (1 extra credit point)

0. (a) Name the new Attorney General of the United States.

-OR-

(b) What city will host the 2010 Winter Olympics?

1. Given the following screen output from a ccc computer:

```
$ ls -l
```

```
total 0
```

```
-rwx----- 1 fred 6810 23 Jan 29 08:58 file1.c
```

```
-rwx----- 1 fred 6810 8 Jan 29 13:45 file2
```

```
-rwx----- 1 fred 6810 8 Jan 29 13:46 file3.dat
```

(4 pts) a. Give the command line to change the group access of **file2** to read and execute.

(1 pt) b. What is the id of the group that can now read **file2**?

(2 pts) 2a. How is the **sizeof** function used?

(2 pts) 2b. What does it return?


(5 pts) 3. How does **Round Robin** scheduling improve mean response time in a cpu scheduler?

(20 pts) 4. What is the output from this program?

```
float out = 100.0;
float goofy (float y, int i)
{
    float z, x = 2.0;
    return x*y*i;
}
int mixup ( int i, int t[5], float j)
{
    static float tmp2 = 4.0;
    int tmp;
    tmp = 50;
    tmp2 = tmp + tmp2 + i;
    out = 2*out;
    j = i + 1.0;
    i++;
    t[i+1]++;
    printf("m: %2d %6.3f %2d %6.2f out = %6.2f\n",
        i, j, tmp, tmp2, out);
    return t[i-1];
}
int main ()
{
    int i, j, tmp, s;
    int table[5] = {2,4,3,1,5};
    float x, y, z;

    tmp = 16;
    i = 1;
    x = 1/2;
    y = 8.0*i;
    z = y/5;
    printf("%6.2f %6.2f %6.2f\n", x, y, z);
    for (j = 0; j < 2; j++)
    {
        s = mixup(i, table, y);
        out = out + s - j;
        tmp = goofy(y,table[j]);
        printf("M: %d %d %d %d %f \n",
            i, s, j, tmp, out );
        i++;
    }
    printf("T:");
    for (j = 0; j < 5; j++)
        printf(" %d", table[j]);
    return 0;
}
```

Output Box

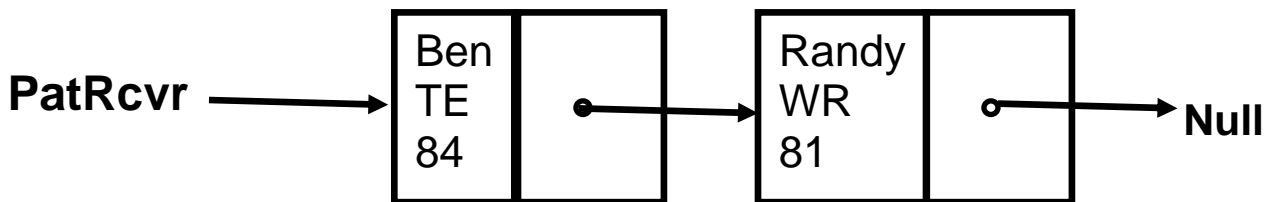


(15 pts) 5. What is the output from this program?

```
int main ()
{
  int x[2] = {11};
  int *pptr, *qptr;

  qptr = (int *) malloc(sizeof (x));
  *qptr = x[0] - 55;
  x[1] = x[1] + 77;
  pptr = (int *) malloc(sizeof (int));
  pptr = qptr;
  printf ("%d %d %d\n", x[0], *pptr, *qptr);
  qptr = (int *) malloc(sizeof (x));
  qptr = &x;
  printf ("%d %d %d\n", x[1], *pptr, *qptr);
  pptr = qptr;
  qptr = (int *) malloc(sizeof (x));
  pptr++;
  *pptr = 88 + x[1];
  qptr = (int *) malloc(sizeof (int));
  printf ("%d %d %d \n", x[1], *pptr, *qptr);
  free(qptr);
  return 0;
}
```

Output Box



(16 pts) 6. Complete this C program such that the linked list **PatRcvr** looks like the diagram above when the program finishes.

```
#include <stdio.h>
typedef struct{
    char *name;
    char *position;
    int number;
    struct Player *pptr;
} Player;

typedef Player *Pptr;

const char *names[] = {"Tom", "Randy", "Ellis", "Ben"};
const int nums[4] = {12, 81, 27, 84};
const char *pos[] = {"QB", "WR", "CB", "TE"};

Pptr createnode (int index)
{
    Pptr newPtr;

    newPtr = malloc(sizeof(Player));

    newPtr->name = names[index];
    newPtr->position = pos[index];
    newPtr->number = nums[index];
    newPtr->pptr = NULL;

    return newPtr;
}
int main()
{
    Pptr PatRcvr = NULL;
    Pptr RcvPtr;
    int i;

    return;
}
```

(18 pts) 7. Complete the function `innersquare` that has as the function prototype

**`innersquare ( char figure[SIZE ] [SIZE], char startchar)`**

**`innersquare`** checks to see if **`SIZE`** is large enough to fill the figure array with three concentric squares spaced three blanks apart in figure (see the diagram below). If **`SIZE`** is large enough, **`innersquare`** starts with the outermost border and fills the outer square of figure with **`startchar`**. **`Innersquare`** then fills a second square that is three blank rows and columns inside the outer square with the next character in ASCII after **`startchar`** (assume `startchar` has already checked outside the function to be between 'a' and 'q'). Finally **`innersquare`** fills a third square that is three blank rows and columns inside the second square with the character after the one placed in the second square.

As an example assume the call to the function is inside this code snippet:

```
#define SIZE 40
```

```
int BLANK = ' ';
```

```
int main {  
char field [SIZE][SIZE]
```

```
...  
innersquare( field, 'f')
```

```
}  
int innersquare (char figure[SIZE][SIZE], char schar)
```

```
{  
    int i,j;  
    for (i=0; i < SIZE; i++)  
        for (j=0; j < SIZE; j++)  
            figure[i][j] = BLANK;
```

```
}
```

