CSC 230 Section 001
Second (Optional) Midterm test
April 9, 2007

Questions Q1 and Q2 carry 20 marks each, all other questions carry 10 marks each. This adds up to 80, but 75 is a perfect score. If you obtain more than 75, your marks will be added up to 75 and will be considered 100%.

It should be possible to answer the questions in the space provided. Use the back of the paper where you are specifically asked to. Otherwise, do not use the back of the paper or attach extra sheets unless you absolutely have to. You may consult any books, notes, etc., but no other person. You must not use a cellphone, computer, PDA, or any other information processing device.

This is a one hour test.

First Name: _____________________________
Last Name: _____________________________
Student ID: _____________________________

Pledge of Honor: I pledge on my honor that I have answered the questions in this test purely on my own individual ability, and have neither given help to nor received help from another person.

Signature: _____________________________
Q1. Consider the following complete program (intended to be treated as a single file and compiled into a binary executable). In the table below, write down the values printed by the program. (If it is not possible to write down the value, indicate why.)

```c
#include <stdio.h>
#include <stdlib.h>
#define F_SWITCH 2

int iG1[10] = {11,12,13,14,15,16,17,18,19,20};
int iG2[10] = {5,4,3,2,1};

void print_element (int *p)
{
    printf ("First: %d\n", *p);
    printf ("Second: %d\n", p[3]);
    printf ("Third: %d\n", *p+4);
    printf ("Fourth: %d\n", *(p+4));
    printf ("Fifth: %d\n", *(p+(p[5]-(p+3))));
    printf ("Sixth: %d\n", *(p+=7));
    printf ("Seventh: %d\n", p[1]);
    printf ("Eighth: %d\n", p);
    printf ("Ninth: %d\n", &p[3]);
}

int main ()
{
    int *p_2_global;

    if (F_SWITCH == 1) {
        p_2_global = iG1;
    }
    else {
        p_2_global = iG2;
    }

    print_element (p_2_global);
    return (EXIT_SUCCESS);
}
```

Printouts:

| First: 5 |
| Second: 2 |
| Third: 9 |
| Fourth: 1 |
| Fifth: 3 |
| Sixth: 0 |
| Seventh: 0 |

Not possible to answer: location of wherever global array happens to be
Same as previous one
Q2. The following code segment does some string manipulation with pointers. Assume that the memory allocations are successful. For every line of code after, state what the contents of each of the 20-character buffers pointed to by s1 and s2 will be. For example, after the strcpy calls immediately following the memory allocations, the contents will be as given below.

```c
char *s1, *s2, *s3, *s4;
int i;

s1 = calloc (20, sizeof (char)); s2 = calloc (20, sizeof (char));
strcpy (s1, "copper cable"); strcpy (s2, "optical fiber");

s1->c o p p e r c a b l e \
\0 \0 \0 \0 \0 \0 \0 \0
s2->o p t i c a l f i b e r \0 \0 \0 \0 \0 \0 \0 \0

*(s1+7) = 'f';

s1->c o p p e r f a b l e \
\0 \0 \0 \0 \0 \0 \0 \0
s2->o p t i c a l f i b e r \0 \0 \0 \0 \0 \0 \0 \0

for (i=8; i++) {
    s1[i] = s2[i+1];
    if (s2[i+1] == '\0')
        break;
}

s1->c o p p e r f i b e r \0 \0 \0 \0 \0 \0 \0 \0
s2->o p t i c a l f i b e r \0 \0 \0 \0 \0 \0 \0 \0

s3 = s1+3; s4 = s2+2;
strcpy (s3, s4);

s1->c o p p e r f i b e r \0 \0 \0 \0 \0 \0 \0 \0
s2->o p t i c a l f i b e r \0 \0 \0 \0 \0 \0 \0 \0

strcat ((s4-2), " link");

s1->c o p p e r f i b e r \0 \0 \0 \0 \0 \0 \0 \0
s2->o p t i c a l f i b e r l i n k \0 \0 \0 \0

for (i=17; i>=13; i--) {
    s3 = s2+i+1;
    *s3 = s2[i];
}
s3[-1] = 'b';

s1->c o p p e r f i b e r \0 \0 \0 \0 \0 \0 \0 \0
s2->o p t i c a l f i b e r b l i n k \0 \0
Q3. Briefly describe what a pointer increment statement such as \( p++ \) achieves, where \( p \) is a pointer to some particular data type. (Use the facing page to write your answer if you need to.)

It updates the variable \( p \) so that it is pointing to the next object after the one it was originally pointing to, where object is the type to which \( p \) was defined as pointing to (not the next byte).

Q4. Consider the following code:

```c
typedef struct struct1 {
    short countrycode;
    short areacode;
    short exchange;
    short number;
} phonenumber;

typedef struct struct2 {
    int height;
    int weight;
    phonenumber *phone;
} person;
```

At some point in the source code, \( p \) is a pointer of type `person*`. Write the pointer expression that returns the areacode part of the phone number of that person (assuming the memory for the data structure is currently allocated correctly). Avoid unnecessary de-referencing operators, and unnecessary parentheses.

\[ p->\text{phone}->\text{areacode} \]

Q5. State in one or two sentences what the function `tolower()` is expected to do. Make sure you describe all cases of the input. (Use the facing page to write your answer if you need to.)

See manual page of tolower; make sure you mention what happens when the argument is not an uppercase character.

Q6. (a) Consider the code below. State with reasons whether it is possible to predict with certainty what will be printed out by the function `func_1`; and if it is possible, state what will be printed. (Use the facing page to write your answer.)

```c
void func_1 (int i, int j)
{
    printf ("i is %d, j is %d\n", i, j);
}
```

```c
/*    . . .  */
/* somewhere in the code, a call to func_1 */
int i = 30;
func_1 (i, i++);
/*    . . .  */
```

It is not possible; the order of evaluation of function parameters is not fixed in C.

(b) In a makefile, the following lines are found:

```make
some_func.o : some_func.c
    gcc -ansi -pedantic -Wall -c some_func.c
```

Briefly describe in English what these lines communicate to the make utility. (Use the facing page to write your answer.)

The target `some_func.o` depends on `some_func.c` (must be re-built if `some_func.c` was modified after the time this target was last built), and can be built by executing the command `gcc -ansi -pedantic -Wall -c some_func.c`