CS1010 Programming Methodology

Week 13: File Processing and Revision

Learning is not attained by chance, it must be sought for with ardor and attended to with diligence. ~ Abigail Adams

To students:

Some programs for this discussion are on the CS1010 website, under the "Discussion" page. Alternatively, you may copy the programs into your UNIX account. For example, to copy Unit19_feof.c, you can type:

cp ~cs1010/discussion/prog/week13/Unit19_feof.c .

File Processing

1. The program **Unit19_feof.c** given in the lecture is shown below.

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    FILE *infile;
    int num;

    if ((infile = fopen("feof.in", "r")) == NULL) {
        printf("Cannot open file \"feof.in\"\n");
        exit(1);
    }

    while (!feof(infile)) {
        fscanf(infile, "%d", &num);
        printf("Value read: %d\n", num);
    }

    fclose(infile);
    return 0;
}
```

If the input file **feof.in** contains the following:

```
10 20 30
```

the output of the program is as follows:

```
Value read: 10
Value read: 20
Value read: 30
Value read: 30
```

Why is it so? How could you correct it?

- 2. Convert your programs for questions 5 and 6 of Discussion Week 12 to read data from a text file (e.g., tiles.in / cap.in) and print the output to another text file (e.g., tiles.out / cap.out). Include statements for handling the error whereby the input file does not exist.
- 3. Merge Sort is a more advanced sorting technique. We are not going to explain how it works here, but one idea employed in Merge Sort is to merge two sorted list into a bigger sorted list.

For instance, given two sorted lists (-3, 8, 65, 100, 207) and (-10, 20, 30, 40, 65, 80, 90), the merged list would be (-10, -3, 8, 20, 30, 40, 65, 65, 80, 90, 100, 207).

Write a program to read two sorted lists of integers from two input text files, merge the lists, and write the merged list to an output text file. You should write a function to merge the lists:

merge(int arr1[], int size1, int arr2[], int size2, int arr3[]);

where arr1 and arr2 are the two given lists with sizes size1 and size2 resepectively, and arr3 is the merged list. You may make your own assumption on the largest size of the lists.

A sample run is shown below:

```
Enter input file for 1st list: q3_list1.in
Enter input file for 2nd list: q3_list2.in
Enter output file for merged list: q3_list3.out
```

The input files and the output file are shown below. The number on the first line of each file indicates the size of the list.

q3_list1.in: q3_list2.in

5	7
-3	-10
8	20
65	30
100 207	40
207	65
	80
	90

q3_list3.out:

· -		
12		
-10		
-3		
8		
20 30		
30		
40		
65		
65		
80		
90		
100		
207		

Revision

4. [10 marks] Study the following C function:

```
int foo(int n) {
   int i, a[3] = {2,3,5};

if (n == 1)
     return 1;

for (i=0; i<3; i++)
     if (!(n % a[i]))
        return foo(n / a[i]);

return 0;
}</pre>
```

(a) [3 marks] Give the sequence of function calls in the order of execution as well as the final return value for the following references:

(b) [3 marks] Briefly describe the functionality of $f \circ \circ$ (). Marks will be deducted for long-winded answers.

(c) [4 marks] Give an iterative version of $f \circ \circ$ (). You must complete the function given below and make use of the 'for' loop.

```
int foo_iter(int n) {
   int i, a[3] = {2,3,5};

for (i=0; i<3; i++) {
      /* Fill in your code here. */
}</pre>
```

5. The content of a two-dimensional array number [6] [6] is given below:

15	22	8	11	59	44
50	64	29	10	34	20
17	86	27	98	57	21
6	16	88	42	36	45
31	26	12	23	82	54
28	66	58	69	41	1

Consider the following code fragment, where selectionSort is the Selection Sort technique covered in lecture.

```
int i, j, tempNumber[6];

for (i = 0; i < 6; i++)
    selectionSort( number[i] );

// At this point, fill in Table (a) on the content in the number array.

for (j = 0; j < 6; j++) {
    for (i = 0; i < 6; i++)
        tempNumber[i] = number[i][j];

    selectionSort( tempNumber );

    for (i = 0; i < 6; i++)
        number[i][j] = tempNumber[i];
}

// At this point, fill in Table (b) on the content in the number array.</pre>
```



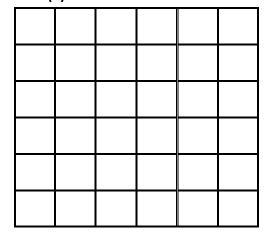
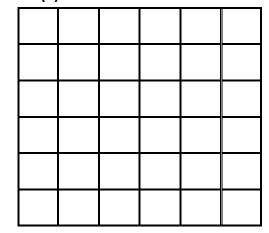


Table (b)



Due to a special property of this sorted number array in Table (b), we can design an algorithm similar to binary search algorithm to search for a key in the array. Below is the recursive algorithm with some parts (4 blank lines) left for you to fill. The function is to be invoked by calling:

```
search( wantedNumber, table, 0, 0, 5, 5 )
```

which returns 1 if wantedNumber is in the number array, or 0 otherwise.