CS 2073 Computer Programming with Engineering Applications

Fall 2009 – Final — Dec 11, 2009

Name:………………………   Score: ……./30

This exam has six questions. You have 120 minutes. Good luck…

1. (5 points) Fill in the blanks and/or answer true (**T**) or false (**F**) questions. Briefly **justify** your answer to get full credit.

   a. (T / F) The result of `(7 / 2 * 2 - 2)` is 5.

   b. (T / F) Suppose currently `i is 3` and `j is 2`. The the following code will print "`ABC`"

   ```
   if( i=5 || j > 3)
     printf("ABC");
   else
     printf("XYZ");
   ```

   c. The following code will print ______ lines.

   ```
   for(i=-2; i < 5; i++)
      if (i/2 < 2) printf("line: i is %d ----- \n", i);
   ```

   d. After the following code, the values in array `a[5]` will be ___________________.

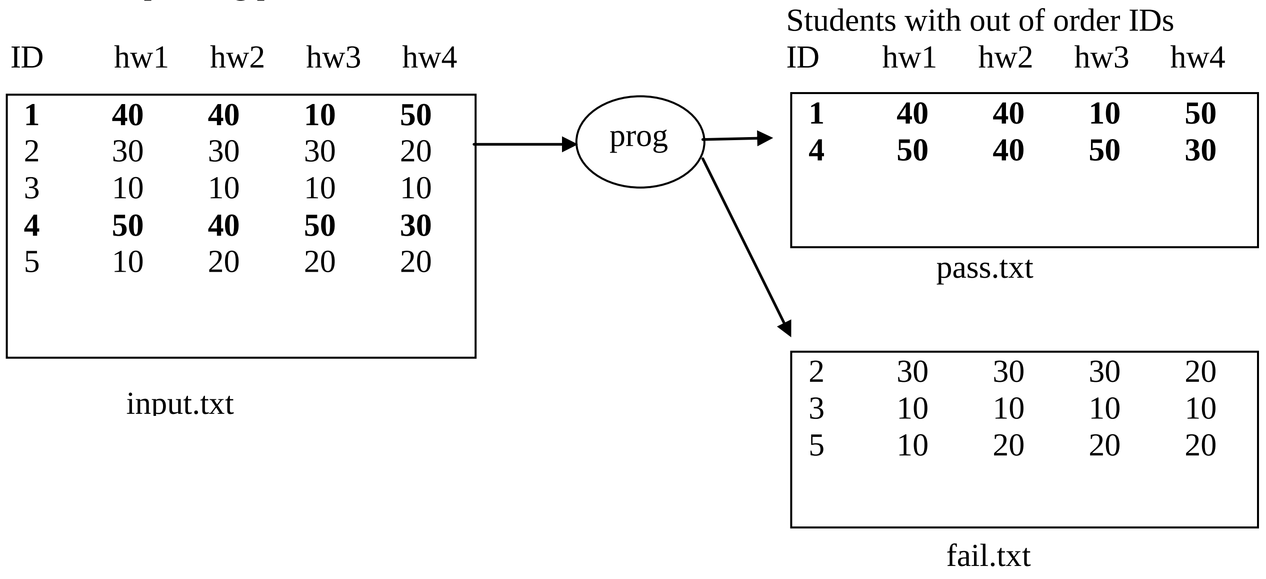
   ```
   int i, a[5] = {4, 2, 5, 6, 8};
   for(i=1; i < 4; i++)
      a[i] =  a[i-1] + a[i+1];
   ```

   e. (T / F) If we pass a parameter to a function using **call-by-reference** technique, then that function **cannot** change the value of actual parameter.

2. (5 points) Suppose we have a data file that stores students' ID and 4 hw grades. So we have 5 columns in each row.

   We would like write a program to divide the students' records into two groups based on their average hw grade.
   - If a student's hw average is equal or greater than 30, then we will print his/her ID and all hw grades into pass.txt file;
   - Otherwise, we will print his/her ID and all hw grades into fail.txt file;

   For instance, your program should process the following input.txt and generate the corresponding pass.txt and fail.txt files:



ID	hw1	hw2	hw3	hw4
1	**40**	**40**	**10**	**50**
2	30	30	30	20
3	10	10	10	10
4	**50**	**40**	**50**	**30**
5	10	20	20	20

input.txt

Students with out of order IDs

ID	hw1	hw2	hw3	hw4
1	**40**	**40**	**10**	**50**
4	**50**	**40**	**50**	**30**

pass.txt

2	30	30	30	20
3	10	10	10	10
5	10	20	20	20

fail.txt

Complete the C program in the next page that reads input.txt and generates pass.txt and fail/txt files as described above.

Name:………………

```
#include <stdio.h>
int main(void)
{
   FILE *infp, *pass, *fail;
   int ID, i, sum, hw[4]={0}; /* if needed declare more variables */


   if( (infp = fopen("input.txt", "r"))==NULL){
      printf("Input file cannot be opened\n");  return 0;
   }
   if( (pass = fopen("pass.txt", "w"))==NULL){
      printf("Output file cannot be opened\n");  return 0;
   }
   if( (fail = fopen("fail.txt", "w"))==NULL){
      printf("Output file cannot be opened\n");  return 0;
   }













   fclose(infp);   fclose(pass); fclose(fail);
   return 0;
}
```

Name:…………………

3. (5 points) Write a function that computes and returns the dot product of two vectors.

BACKGROUND: The dot product of two vectors a = [a_0, a_1, a_2, … , a_{n-1}] and b = [b_0, b_1, b_2, … , b_{n-1}] is defined as:

$$a \bullet b = \sum_{i=0}^{n-1} a_i b_i = a_0 b_0 + a_1 b_1 + a_2 b_2 + \cdots + a_{n-1} b_{n-1}$$

where Σ denotes summation notation and n is the dimension of the vectors.
For example, the dot product of two (n=3)-dimensional vectors [2, 3, −5] and [4, −2, −1] is (2)*(4) + (3)*(-2) + (-5)*(-1) = 7.

```
int dot_product(int a[],int b[], int n)
{




     return …………
}
```

Name:……………………

4. (5 points) Suppose we declare three arays: **`int a[N], b[M], c[N+M]`**; where **`N`** and **`M`** are predefined constants. Assume that the values in **`a[N]`** and **`b[M]`** arrays are already **sorted**. We are now interested in **merging** the numbers in these two arrays and put them in array **`c[N+M]`** while keeping the values in order. For example, if we have

```
int a[5] = {2, 5, 7, 8, 12};
int b[4] = {1, 2, 5, 9};
```

then **`c[9] = {1, 2, 2, 5, 5, 7, 8, 9, 12}`**.

Write a function that will take **`a[], b[], c[], n`** and **`m`** as parameters and find the merged array as described above.

```
void union(int a[], int b[], int c[], int n, int m )
{



 return;
}
```

Name:……………………

5. (5 points) Suppose we are given some terrain data represented by a two-dimensional array, where each cell of the array is an integer number denoting the elevation at this position. We are interested in determining the cells that represent **peaks**. The idea is to check every cell [i][j] that has 8 neighbors surrounding cell [i][j] and see if the value at [i][j] is greater than the values of its 8 neighbors. If so print i and j and the value in cell [i][j] as one of peaks. Note that boundary cells do not have 8 neighbors, so we will not consider them. For example, suppose we are given a 4x6 terrain data as follows,

4	5	3	2	1	0
6	1	6	1	**13**	5
3	**10**	6	5	9	12
6	2	3	12	5	1

Your function should check all the cells in shaded area, and generate the following output:

```
Cell 1,4 is a peak and its value is 13
Cell 2,1 is a peak and its value is 10
```

Write a function that takes **`t[X][Y]`** as a parameter and determines the peaks and their row, column, and values as described above. X and Y are predefined global constants (e.g., X=100, Y=200).

```
void print_peaks(int t[X][Y])
{




  return;
}
```

Name:……………………

6. (5points) Consider the following code, what will be the output of this code? Also show how the values of variables change in memory. To get partial credit, show your work.

```
main()
{
  int x=7, y=5, z=1;

  printf("First: %d %d %d \n", x, y, z);

  z = myfunction( &x, &y, z);

  printf("Second: %d %d %d \n", x, y, z);

}

int myfunction(int *a, int *b, int c)
{
     int tmp;

     c = c + *a - *b;

     *b = 10 - 2 * c;

     *a = 5 + *b / 3;

     printf("In Func %d  %d \n", a, *a);

     tmp = c + *a / *b;

     return tmp;
}
```

name	addr	Memory content
x	105	
y	106	
z	107	
	108	
	109	
a	110	
b	111	
c	112	
tmp	113	
	114	
	115	
	116	
	117	
	118	

OUTPUT