Student Name: (ALL CAPS) ..... NID:.....

COP 3223 Sec 2: Fall '11 C Programming Test 4 (50 points)

1. (14 points)
   This program reads a file from America's Most Wanted. Write a complete program (with all the headings, etc.) that reads in an integer n, and thereafter on n subsequent lines there are three items per line: a string firstName, a string lastName, an integer representing HowManyYears this person has been in the America's Most Wanted file (the firstName and lastName each have a maximum length of 20, including the end-of-string character.) You must prompt for the input filename as a string that is sure to be less than 30 chars long (including the end-of-string char). Open the file, etc., read everything, and find the person with the biggest number HowManyYears. After reading is complete, prompt for an output filename string (again, max 30 chars), open it and write to it **ONLY** the first and last names of the person with the biggest number (you can assume there is only one person with the biggest number HowManyYears.) Remember to close both files before ending, and do not use a struct for this program. DO NOT check for any errors.

Student Name (ALL CAPS):..... NID/PID:.....

2. (18 points)
Write **ONLY** a void function to take three parameters: an `int**` array named PIC1 and two ints named numRows and numCols. Assume numRows and numCols **are equal** and are an even number (not odd). The function should **first** copy the upper right triangular (defined as the part of the picture that is above and to the right of the diagonal joining the upper left corner and the lower right corner) half of the picture into the space currently occupied by the lower left triangular half (see picture). This effect will be achieved by copying the top row into the bottom row, but reversing it left to right, and doing the same for every (incrementally shorter) row of the picture. **Second**, as an added requirement, the function should draw a line (of thickness 4, and brightness 255) immediately to the **left of the right half** (see picture).

Original face picture

Desired output picture

Student Name (ALL CAPS):..... NID/PID:.....

3. A. (8 points) Suppose you have two structures:

```
struct Postal {
        int zip;
        int num_residents;
    };
```

```
struct Births {
        int zip;
        int newborns;
    };
```

The post office uses the first struct to keep track of the number of residents in a zip-code zone. The second is used by the county to keep track of births by the month in each zip-code region.

Write ONLY the function `UpdatePostal` that takes in one `struct Postal` and one `struct Births` (both parameters are pass-by-value); a third parameter is a pass-by-reference `int` parameter named `flag`. The function must return a `struct Postal` via the function's name. If both `zips` agree (one from `Postal` and one from `Births`), the field `num_residents` in the returned struct must be made to be the sum of the old `num_residents` and `newborns`. Additionally, if both agree, set the `flag` to 1 (so it will be available to the caller). The returned struct will have to get the `zip` from parameter one. If the two `zips` do not agree, return the first parameter's data through the function's name, but also set the `flag` to zero.

WRITE THE SOLUTION TO B ON THE BACK OF THIS PAGE.
B. (7 points) The system also has a struct definition for deaths

```
struct Deaths { int zip; int died;};
```

Write ONLY the function `ReducePostal` that takes in one `struct Postal` and one `struct Deaths` (the first parameter is pass-by-reference the second is pass-by-value). If both `zips agree` (one from `Postal` and one from `Deaths`), the field `num_residents` in `Postal` must be reduced by the number `died`; this reduction will be available to the caller. Additionally, if both agree, the function returns the `int 1`. If the two `zips` do not agree, return the `int` 0, and leave the values in the first parameter unchanged.

Student Name: (ALL CAPS) ..... NID:.....

4. (6 points) What is the program output caused by the second, fourth, and seventh call to `Print_Array`? (This is the same code you have seen in the notes, only the 10 actual numbers are changed).

```
#include <stdio.h>
#include <stdlib.h>
#define         SIZE    10
void Print_Array(int values[], int length);
void swap(int values[], int i, int j);
void Move_Max(int values[], int max_index);
void Simple_Sort(int values[], int length);
int main() {
  int my_vals[SIZE] = {34, 94, 64, 12, 76, 34, 30, 85, 67, 45};
                                  /*************************************/
  Print_Array(my_vals, SIZE);     /* <-- this is the  FIRST call to Print */
                                  /*************************************/
  Simple_Sort(my_vals, SIZE);
}
void Simple_Sort(int values[], int length) {
  int i;
  for (i=length-1; i> 0; i--)
  {
    Move_Max(values, i);
    Print_Array(values, SIZE);
  }
}
void Move_Max(int values[], int max_index) {
  int max, i, maxi;
  max = values[0];
  maxi = 0;                                  WRITE YOUR FINAL ANSWERS HERE
  for (i=1; i<=max_index; i++)              ---------------------------------------
  {                                 second |
    if (max < values[i])                   |---------------------------------------
    {                                      |
      max = values[i];              fourth |
      maxi = i;                            |---------------------------------------
    }                                      |
  }                                 seventh |
  swap(values, maxi, max_index);           |---------------------------------------
}
void swap(int values[], int i, int j) {
  int temp;
  temp = values[i];
  values[i] = values[j];
  values[j] = temp;
}
void Print_Array(int values[], int length) {
  int i;
  for (i=0; i<length; i++)
    printf("%d ", values[i]);
  printf("\n");
}
```