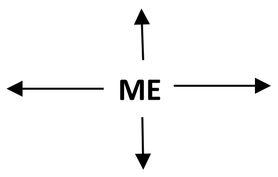
Name [SURNAME, FIRSTNAME]:_________________________________ Section:________
Teacher: ___________________________________

I	
II	
III A	
B	
IV	
V	
VI	
VII	
VIII	
TOTAL	**/90**

CCPROG2 Midterm Exam
ANSWER SHEET



Write the SURNAME of your seatmates:

I. 1D Array Analysis – Array Indexing Notation [10 pts]

1. `char nArray[4] = {'R', 'E', 'A','D'};`	**4.** `E` `(OR: 'E' )`	**7** `fOne(nArray, SIZE);` `(OR: fOne(nArray, 5);)`
2 `R`	**5.** `DEAR`	**8-9. (2pts)** `5.80 7.00 7.00 12.10 22.00`
3. `INVALID`	**6.** `40`	**10.** `6.40 (OR: 6.4)`

II. 1D Array Analysis – Pointer Notation. [10 pts]

1. `28`	**2.** `0.0`	**3.** `7.5`	**4.** `22C0`	**5.** `list`
6. `list, SIZE (OR: list + 0, 7)`		**7.** `list+2 , 3`		
8. `ptr = list+6;` `(OR: ptr = list + SIZE - 1; )`		**9.** `float`	**10.** `float *`	

III. 1D Array Debugging/Analysis.

A. [10 pts]

Line Number	Answer
Line 1	`#include <stdio.h>` (part of given, 0 pt)
Line 6 (2 pts)	`int temp = *a;`
Line 14 (2 pts)	**CORRECT** *(Reason: changing it to a loop starting at the end of the array is* ***not*** *accepted; if it's correct, no need to rewrite as per instructions)*
Line 18 (2 pts)	`if (A[max] < A[j])` `(OR: if (A[j] > A[max]) )`
Line 22 (2 pts)	`Swap(&A[i], &A[max]);` `(OR: Swap(A + i, A + max); )`
Line 29 (2 pts)	`SelectionSort(arr, ARRSIZE);` `(OR: SelectionSort(&arr[0], ARRSIZE);` `OR: SelectionSort(&arr[0], 6);` `OR: SelectionSort(arr, 6);)`

B. [5pts]

1. (1pt) **5**
2. (2pts) **4**
3. (2pts) **5**

IV. 2D Array Analysis [10 pts].

1. `15`	**6.** `func1`
2. `150`	**7.** `func2, func3`
3. `1`	**8.** `Yes` `(OR: Valid , True)`
4. `1`	**9.** `Yes` `(OR: Valid , True)`
5. `15`	**10.** `Yes` `(OR: Valid , True)`

V. Analysis on Strings [10 pts].

1. (3pts) `SHUTDOWN`
2. (3pts) `PRETTYSAVAGE`
3. (4pts) `BLACKPINK`

VI. 1D Array/Strings Programming [10 pts].

1. `<string.h>`	6. `strReverse[strSize] = '\0';` OR: `strReverse[strlen(str)] = '\0';`
2. `int strSize`	7. `strcmp`
3. `strSize - 1` (OR: `strlen(str)-1`)	8. `0`
4. `strSize - 1 - i` (answers for #4 and	9. `strlen`
5. `i` #5 can be interchanged)	10. `19`

VII. Programming with 2D Arrays -1 [10 pts].

1. (1.5pts) dest[x] = source[j][i]; (OR: *(dest+x) = source[j][i];) (OR: dest[i*MAX_R + j] = source[j][i];)	6. a1D
2. (1.5pts) dest[i][j] = source[x]; (OR: dest[i][j] = *(source+x);) (OR: dest[i][j] = source[i*MAX_C + j];)	7. MAX_R * MAX_C (OR: 50)
3. MAX_C * MAX_R (OR: 50)	8. aData
4. a1D	9. a1D
5. aData	

VIII. Programming with 2D Arrays -2[15 pts].

```
/* Assume MAX_ROWSIZE and MAX_COLSIZE are defined constants */
```

```
/* This function getInputsForMatrix() will
get input for nRows number of rows and nCols
number of columns of the 2D array. Only valid
values should be stored.  A valid value is
any integer from -1000 to 1000 (both
inclusive). As long as an invalid value is
given by the user, the user has to give a new
value to replace the invalid value. Storing
into the array should be via column-major
accessing.
*/
void
getInputsForMatrix(int matrix[][MAX_COLSIZE],
                   int nRows, int nCols)
{int i, j;

 for (j = 0; j < nCols; j++)
   for (i = 0; i< nRows; i++)
     do
     {  scanf("%d", &matrix[i][j]);
     }while (matrix[i][j] < -1000 ||
             matrix[i][j] > 1000);
}
```

```
/* This function processOutputs()
accesses the 2D array using ROW-MAJOR
ORDER as it computes for sum of even-
valued elements, sum of odd-valued
elements, total count for number of even
numbers, total count for number of odd
numbers, and total number of zeroes
(exactly equal to 0).  Results of these
are displayed before the end of the
function.
*/
void
processOutputs(
             int matrix[][MAX_COLSIZE],
              int nRows, int nCols    )
{   int r, c, sumE, sumO,
             numE, numO, zero;
    sumE = sumO = numE = numO = zero = 0;

    for (r = 0; r < nRows; r++)
       for (c = 0; c < nCols; c++)
       {    if (matrix[r][c] %2 == 0)
            {    sumE += matrix[r][c];
                 numE++;

                 if (matrix[r][c] == 0)
                    zero++;
            }
            else
            {  sumO += matrix[r][c];
               numO++;
            }
       }
    printf("The sum of evens is %d.\n",
             sumE               );
    printf("The sum of odds is %d.\n",
             sumO               );
    printf("%s %d.\n",
           "The count of even numbers is",
             numE              );
    printf("%s %d.\n",
            "The count of odd numbers is",
             numO              );
    printf("%s %d.\n",
            "The count of zeroes is",
             zero              );
}
```