EECE 231 — Introduction to Programming Using C++ and MATLAB, Sections 1,2,6,7, and 8
Final Exam

Dec 20, 2014

- The duration of this exam is 2 hours and 40 minutes.
- **Submission instructions:**
  - Allow at least **10 minutes** at the end of the duration of the exam to **submit** your answers.
  - **Create a directory under "Volume D:" on your computer and call it after your AUBNet id.**
  - **Place your cpp and m-files in the newly created directory to be collected by the submission collection script.**
  - It is your responsibility to make sure your files are correctly submitted in the designated directories. **Failure to follow the above instructions, for any of the problems you worked on, will result in a failing grade on the problems with missing solutions.**
- The exam consists of **3 problems.** with a total of **10 parts**.
  Total number of points 160. With 15 additional Bonus points.
- Below is a rough ordering estimate of the problems according to "difficulty":
  - Level 1 (60 points): Prob1a (15pts), Prob1c (15pts), Prob1d (15pts), Prob3a (15pts)
  - Level 2 (55 points): Prob1b (15pts), Prob2a (15pts), Prob2b (15pts), Prob3bNonefficient(10pts)
  - Level 3 : Prob2c (15 pts)
  - Level 4 (25 points): Prob2dNonefficient (15 pts), Prob3bEfficient (10 pts + 5 bonus)
  - Level 5 : Prob2dEfficient (5 pts + 10 bonus)
- Plan your time wisely. Do not spend too much time on any one problem. Read through all of them first and attack them in the order that allows you to make the most progress.
- You can use all the material available on "Volume D:" in the `EECE231Files` folder.
- You are **NOT** allowed to use the **web**. You are not allowed to use **USB's** or files previously stored on your machine.
- If you get caught violating the above rules or if you communicate with a person other than the exam proctors during the exam, you will immediately get zero and you will be referred to the appropriate disciplinary committee.
- Active cell phones and any other unauthorized electronic devices are absolutely not allowed in the exam rooms. They should be turned off and put away.
- Submit your solutions each part in a separate file as indicated in the booklet. Include your name and ID number in each file.
- Good luck!

Problem 1 (60 points). Jam

a) **Plot (15 points).** Plot the functions $cos(x)$, $cos(2x)$ and $cos(3x)$ for $0 \leq x \leq 2\pi$ on the same figure using 3 different colors. Use the `plot` function.

Submit your solution in a file called `Prob1a.m` including your name and ID number in the comments.

b) **Interpolation (15 points).** Consider the real valued function

$$f(x) = a_1 \times \cos(x) + a_2 \times \cos(2x) + a_3 \times cos(3x),$$

where a_1, a_2, and a_3 are unknown coefficients. We know that $f(1.1) = 1$, $f(3) = -1.5$ and $f(6) = 3.2$. Write a Matlab script to determine the values of a_1, a_2 and a_3.

Submit your solution in a file called `Prob1b.m` including your name and ID number in the comments.

c) **Symbolic computations (15 points).** Consider the function $f(x) = x^{\cos(x)}$. Use the Matlab symbolic toolbox to plot the third-order derivative $g(x) = \frac{d^3}{dx^3} f(x)$ and determine its value at $x = 1.2$ (you should get 0.9469). Use the `diff`, `ezplot` and `subs` functions. Submit your solution in a file called `Prob1c.m` including your name and ID number in the comments.

d) **Files (15 points).** Create a text file called "data.txt" consisting of integers each on a separate line.

Write a Matlab script which reads the integers in the file "data.txt" into an array A using the `textread` function, sorts the array A using the `sort` function, and then displays the integers in sorted order.

Submit your solution in a file called `Prob1d.m` including your name and ID number in the comments.

For instance, running your script on the "data.txt" file shown below, should produce the following output.

contents of file "data.txt"	output after running the script
10 2 3 7 5 12 6	>> Prob1d 2 3 5 6 7 10 12

Problem 2 (65 points + 10 bonus points). Zeros

a) **All-zeros check (15 points).** Write a function `isVectorZeros(V)` that takes a vector V, returns 1 if all values in V are zeros, and returns 0 otherwise. Any solution is worth full grade.

Examples:

```
>> isVectorZeros([0 0 0 0])
ans =
     1
```

```
>> isVectorZeros([0 0 2 0])
ans =
     0
```

b) **Number of Zeros (15 points).** Write a function `numOfZeros(A)` that takes a two dimensional matrix A, and returns the number of zero entries in A. Any solution is worth full grade.

Examples:

```
>> numOfZeros([0 0 ; 2 0])
ans =
     3
```

```
>> numOfZeros([0 0 2 0; 3 2 0 0; 0 0 0 1])
ans =
     8
```

c) **Maximum number of zeros per row (15 points).** Write a function `maxZerosRow(A)` that takes a two dimensional matrix A, and returns the maximum number of zeros per row.

Any solution is worth full grade.

Examples:

```
>> maxZerosRow([0 2 0; 2 0 1])
ans =
     2
>> maxZerosRow([1 1 0; 0 2 0; 2 0 1])
ans =
     2
```

```
>> maxZerosRow([1 0; 0 2 ])
ans =
     1
>> maxZerosRow([1 3; 2 2 ])
ans =
     0
```

d) **Maximal-width all-zeros square sub-matrix (20 points + 10 bonus points).** Write a function

```
[i,j,w] = maxZeroSquare(A)
```

that takes a two dimensional matrix A of size $m \times n$ and finds a **maximal-width square sub-matrix** of A consisting entirely of zeros. The function should return the coordinates (i, j) of the top left corner and the width w of the square.

That is, all the entries of the returned square sub-matrix of A should be zeros and all other squares sub-matrices of A whose entries are all zeros should have width less than or equal to w. If no such square sub-matrix exists, the function should set i, j and w to -1.

Examples:

```
>> A = [

 1  0  1  1
 1  0  0  1
 2  0  0  2
 1  0  2  2
 1  1  1  3

];
>> [maxi,maxj,maxw]=maxZeroSquare(A)
maxi =
     2
maxj =
     2
maxw =
     2
```

```
>> A = [

 1  2
 0  1

];
>> [maxi,maxj,maxw]=maxZeroSquare(A)
maxi =
     2
maxj =
     1
maxw =
     1
```

```
>> A = [

   1  0  0  0  0  1  1  0
   0  1  0  0  0  0  0  0
   0  0  0  0  0  0  0  0
   0  0  0  0  0  0  0  0
   1  0  0  0  0  1  2  1
   1  0  0  0  0  0  0  0
   0  0  1  0  0  0  0  0
   0  2  3  0  0  1  1  0

];
>> [maxi,maxj,maxw]=maxZeroSquare(A)
maxi =
     3
maxj =
     2
maxw =
     4
```

Any correct solution is worth 15 points. Faster solutions are worth more points.

Write a script to test your functions and save it in a file called `Prob2.m`. Save you functions in `m-files` with the corresponding names.

Problem 3 (35 points). C++

a) **Volume of a cone (15 points).** The volume of a cone is given by the formula $V = \frac{1}{3}\pi r^2 h$, where r is the radius of the base circle and h is the height. Write a C++ program which prompts the user for the values of r and h. Your program should compute and print the value of V. Submit your solution in a file called `Prob3a.cpp` including your name and ID number in the comments.

b) **Find local maximum(20 points).** Let $A[0 \ldots n-1]$ be an array of integers of size $n \geq 3$. A *local maximum* of A is an entry $A[i]$ of A, for some $1 \leq i \leq n-2$, such that $A[i] > A[i-1]$ and $A[i] > A[i+1]$.

We will assume in this problem that all contiguous elements in the array are distinct (i.e., $A[i] \neq A[i-1]$ for $i = 1, \ldots, n-1$). We will assume also that $A[0] < A[1]$ and $A[n-2] > A[n-1]$, which guarantees that there must be at least one local maximum in A.

Examples:

- The local maximum of the array $A[5] = \{1, 2, \mathbf{3}, 2, 0\}$ is 3.
- The local maxima of the array $A[11] = \{1, 2, \mathbf{3}, 2, 0, -1, 5, 7, \mathbf{8}, 4, 2\}$ are 3 and 8.
- The local maxima of the array $A[13] = \{1, 2, \mathbf{3}, 2, 0, 3, \mathbf{4}, -1, 5, 7, \mathbf{8}, 4, 2\}$ are $3, 4$ and 8.

Write a function

```
int findLocalMaximum(int A[],int n)
```

which given A and its size n, finds a local maximum of A. If A has more than one local maximum, any local maximum of A is a valid answer. Assume in your solution that $n \geq 3$ and that $A[0] < A[1]$ and $A[n-2] > A[n-1]$.

Write a program to test your function.

Any correct solution is worth 10 points.

To get full grade do it efficiently (much faster than linear time). Note that the assumptions "$A[0] < A[1]$ and $A[n-2] > A[n-1]$" is not needed in the slow solution. Use it in the fast solution. Submit your solution in a file called `Prob3b.cpp` including your name and ID number in the comments.

Best of luck and happy holidays!