Computer Lab Assignment 2

Pixel Graphics and Ocean Volume

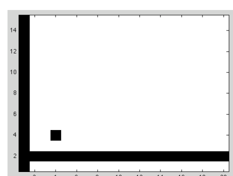
This lab has two parts. In the first part, we learn to fill matrices and display their contents graphically. In the second part, we load a data file with the topography of the Earth and calculate the volume of the oceans.

(1) For this lab, a few more Matlab instructions are needed but we do not want to retype them each time. Instead we put them into a script file e.g. *example.m*. The extension *.m* labels it as a Matlab file. Please open the editor and type in the following commands:

```
maxX = 20;
maxY = 15;
A=zeros(maxX,maxY);
A(1,:) = 1
A(:,2) = 1
A(4,4) = 1
%the following lines just display the matrix. The details are not important!
imagesc(A');
set(gca,'YDir','normal');
colormap(1-gray);
axis equal;
axis tight;
```

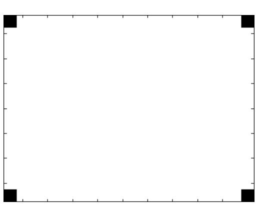
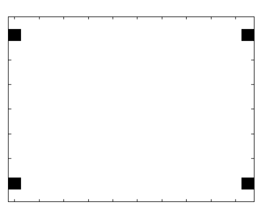
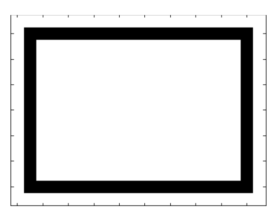
Now save the file, go back to the command window, and execute the file by typing

```
>> example
```

you should see the following image appear: 

How does this work? The graphics command translates the contents of matrix A into pixels. A **zero** at point (i,j) represents **a white pixel**. **1 means black**. Now look how the indices work. What is the first and what is the second index?

(2) No modify the file above to generate the following images. This will take some time.

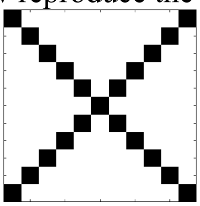
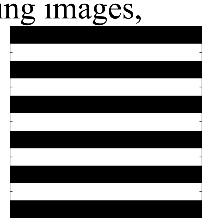
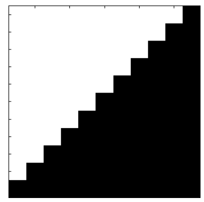
If you can, please write your code so that it works for arbitrary `maxX` and `maxY`.

(3) Now will use a *for* loop:

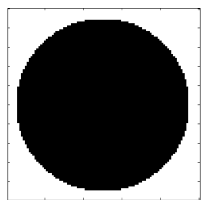
```
maxX = 11;
maxY = 11;
A=zeros(maxX,maxY);
for i=(1:maxX)
   A(i,i) = 1;
end
```

Then add the five graphics commands from above. What image do you get?

(4) Now reproduce the following images,

(5) Now increase the resolution for one final image:



It requires you to write two *nested* loops

```
for i=(1:maxX)
    for j=(1:maxY)
        (add your own stuff here)
    end
end
```

This concludes part 1. Please turn over.

PART II

We now want to apply our skills to compute the combined volume of all oceans on this planet. Therefore we load a file into Matlab that contains the topography *H(latitude, longitude)* in matrix form:

	0	20	40	60	80	100	120	140	longitude		200	220	240	260	280	300	320	340
85N	-4298	-4317	-4326	-4321	-4299	-4262	-4233	-3980	-3885	-3885	-3859	-3655	-3380	-3459	-3350	-3642	-4132	-4242
75N	-2859	122	-273	-168	-100	-61	-3263	-1351	-1877	-1489	-3162	-3773	-1884	-312	1039	1317	2175	148
65N	-3192	426	-132	32	71	510	258	360	32	-41	336	209	-132	-60	-16	-1104	2957	-769
55N	-117	-64	148	132	90	315	374	573	109	-2914	383	133	552	303	-79	-2050	-2926	-2741
45N	72	582	147	233	508	1668	780	44	-5683	-4988	-4927	-3856	1155	420	288	-213	-4371	-4208
35N	-511	409	1894	139	1219	1792	-23	370	-5495	-5495	-5489	-5133	1548	633	641	-5114	-4708	-4874
25N	373	123	741	1656	1919	3920	330	-2530	-5853	-4858	-5657	-4692	-3505	404	-748	-5578	-3830	-4578
15N	401	427	-63	-3286	355	683	-3273	-4701	-4735	-3975	-4571	-5281	-4032	2626	-3437	-5041	-5250	-3216
5N	234	415	1076	-4092	-72	-36	-1570	-2274	-3849	-5999	-4392	-4925	-4304	-3766	-529	-539	-3719	-4310
5S	-4600	366	118	-4560	-4605	487	-1104	-3158	-2979	-5281	-4829	-4448	-4183	-3266	768	137	-3696	-4334
15S	-5552	1133	-565	-239	-5046	-4898	-844	-43	-3662	-3202	-4940	-3784	-3892	-4273	-3308	315	416	-4993
25S	-5372	1124	-2787	-3896	-5244	-6021	148	146	-1170	-2383	-4703	-3210	-3548	-4144	-4163	144	-1275	-4693
35S	-3173	958	-4764	-3787	-4004	-3746	434	52	-1403	-2326	-5276	-4285	-3609	-3605	-3958	48	-4026	-4141
45S	-4849	-5001	-3440	-5295	-3086	-4041	-4946	-5070	-4724	-3012	-5093	-5316	-4499	-4333	-4250	-12	-5196	-3536
55S	-3116	-4388	-3838	-4550	-3740	-3574	-4176	-3809	-4341	-4598	-5003	-4411	-3174	-4319	-3866	-138	-3934	-4325
65S	-5334	-5168	-5244	-4851	-2022	-4526	-4381	-4370	-3085	-4397	-3451	-3947	-4978	-5162	-4952	-3934	-2739	-3742
75S	-198	67	1513	2461	2362	2634	2663	2302	694	-3017	-4359	-4366	-2784	-2827	-582	-348	-4226	-4482
85S	2690	3434	3621	3500	3901	3452	3074	2358	466	0	0	590	1490	2165	571	129	13	2212

The columns and rows correspond to values of longitude and latitude. The values represent the elevation of the ground *in meters* relative to sea level. *Positive* value imply mountains, *negative* values are presumed to give the depth of the ocean at this point on the planet.

Files with two different levels of resolution are provided on bCourses:

`topography_18x18_grid.txt`
`topography_180x360_grid.txt`

Start a new Matlab file and load in the first data set:

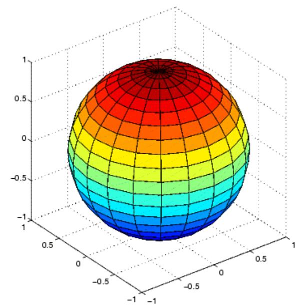
```
H = load('topography_18x18_grid.txt')
size(H)
ny = ...
nx = ...
R = 6371000; % radius of the Earth in meters
area = 0     % keep track of surface area
for iy = 1:ny
    latt = ...
    for ix = 1:nx
        long  = ...
        patch = ...
        area  = area + patch;
    end
end
%Print the calculated area
area
```

Before calculate the volume of the ocean let us derive the surface area of the earth (within the resolution of the grid). You need to do three things to fill in the blanks …

(1) Fill in the number of grid points *nx* and *ny*.

(2) Set the values of *longitude* and *latitude* depending on the variables *ix* and *iy*. *ix* and *iy* run over all elements of the matrix.

(3) Determine the size of the patch depending on latitude, longitude, *nx*, and *ny*. Run the code and test if you approximately reproduce the analytical value $A=4\pi R^2$. This picture may help you determining the patch size:



(4) Now we actually want to compute the fraction of the Earth's surface that is covered by water. This requires an *if* statement because we only want to add those area where the elevation is below zero. Compute this fraction for the file `topography_18x18_grid.txt`, then switch to `topography_180x360_grid.txt` and determine it again more accurately. *(Please avoid the common mistake of switching latitude and longitude indices in the matrix. There are 360 longitude intervals but only 180 latitude steps.)*

(5) Now compute the volume of the oceans by adding height times area for all patches where there is water.