Matlab Programming Summaries

Part I

Getting inputs

```
A= input('Enter the value of A:');
```

Output strings

```
str=['the value of pi=' num2str(pi)];
disp(str);
```

Output strings and numbers

```
fprintf('The value of PI is %f\n',pi);
fprintf('The value of PI is %1.5f\n',pi);
```

%d, %e (exponential), %f,%g (float or exponential, whichever is horter),\n

Saving Data

```
save my
```

```
save my A B C
```

```
save my.dat x –ascii
```

First parameter to save command is the filename and the remaining (A B C) are the list of variables which are going to be saved.
Note: Saving data in ascii file will not preserve the variable names.

Loading data

```
load my
```

```
load my a
```

```
load my.dat x –ascii
```

Plot graphs in color, and line styles

Colors: y,m,c,r,g,b,w,k

Marker styles: . o x + * s d v ^ v < > p h

Line style: - : -. –

Example : `plot(x,y,'r-', x1,y1,'bo');`

axis([minx maxx miny maxy]);

legend('first graph','second graph', …, 'position')
position: TL TC TR ML MC MR BL BC BR …

Using Plot Browser

Programming example

Write a program which gets, voltage and resistance and gives out the current and power (I=V/R and P=I^2*R)

Comparison and Logic Operators

```
If(b<2)
…
elseif (b>2)
…
else
…
end
```

Programming example

Write a program to solve 2nd order equation and mention whether roots are complex or real.

Catching errors

```
A=[1 2 3];
try
  index=input('which item you want: ');
  fprintf('Your item: %f', A(index));
catch
  disp('You entered a wrong index');
end
```

Debugging

Setting break points (in code editing window, click on the left hand side bar to put break points)

Using step and continue (use the step and continue buttons on the toolbar to control running process of program lines).

While loop

```
while condition
  …
end
```

Example: write a program to get n numbers and print average

For loop

```
for ii=1:10
end

for ii=1:2:10
end

for ii=[ 5 9 8]
```

```
end
```

Example: Write a program to calculate factorial of a number n

Break…continue

Break command will stop a loop and transfer the execution to the first command after the loop's end. We normally execute break command under a specific condition (using if command).

```
i=0;
a=1;
while i<=10
 if(a>10)
   break;
 end;
 i=i+1
 a=i*2
end
```

Continue command will instead stop running that particular round of loop and continue with running next round of the loop.

Switch… case

```
switch(ii)
case {1, 3 4},
…
case 2,
…
end
```

Programming hints

Use vector type operators in the functions

Arrays

```
a=1:4;
```

Creates an array equal to [1 2 3 4]. The expression a(1) refers to first cell and a(4) to the last cell.

```
a(8)=5;
```

Above command assigns a value of 5 to cell number 8 of the array. Since the array is only 4 cells length, its length is first increased to 8 cells and then the assignment takes place. The outcome is an array which contains [1 2 3 4 0 0 0 5].

Logical arrays

```
A=[1 2 3; 4 5 6; 7 8 9]
B= A > 5;
```

The above command is a array logical "bigger than 5" operation. Each member is compared to 5 and a new array containing logical result of the comparison is built.

```
B= [0 0 0; 0 0 1; 1 1 1];
```

Now above logical array can be used to mask an operation on the first array.

Example: Add 1 to members which are bigger than 5 in the array A.

```
A=[1 2 3; 4 5 6; 7 8 9];
B= A > 5;
A(B)=A(B)+1;
```

The last command takes array B, and runs the above increment only on indices of A on which array B has a value of 1.

The output will be [1 2 3; 4 5 7;8 9 10].

Measuring run time

In order to measure the run time of specific commands in a program “tic” and “toc” commands can be used.

```
a=[1 2 3 ; 4 5 6; 7 8 9];
tic
b=inv(a);
toc
b
```