9. The calculateGrades script is pasted below

```
% calculateGrades
% load a matrix of student grades, normalize each assignment to have a b
% average and then calculate the average grade for each student

% load the grades load
classGrades

% look at the first 5 rows namesAndGrades(1:5,:)

% get just the grades out grades=namesAndGrades(:,2:end);

% calculate the mean of each assignment meanGrades=mean(grades);

% show the meanGrades meanGrades

% calculate nanmean grades and nanmax grades meanGrades=nanmean(grades)

% compress the range so the mean is 3.5
meanMatrix=ones(size(grades,1),1)*meanGrades
curvedGrades=(grades./meanMatrix)*3.5; nanmean(curvedGrades)
curvedGrades(find(curvedGrades>5))=5;

% calculate the total grade for each student
totalGrade=ceil(nanmean(curvedGrades,2));
letters='FDCBA'; letterGrades=letters(totalGrade);
  disp(['Grades: '
letterGrades]);
```

Screen output:

ans =

1.0000	2.5064	3.6529	2.4617	3.3022	2.5189	0.0963	4.6502
2.0000	2.1586	3.2324	3.4737	0.2378	2.4480	0.4194	1.9951
3.0000	4.9878	NaN	4.8637	1.7439	4.3852	4.8740	0.2370
4.0000	4.0580	1.9914	1.6388	2.2567	1.7657	3.2567	1.7119
5.0000	2.4283	3.7491	4.1890	NaN	2.2472	1.1562	3.6798

meanGrades =

NaN NaN 2.8361 NaN 2.8540 1.6481 NaN meanGrades
=

2.9690 2.9445 2.8361 2.4879 2.8540 1.6481 2.5677 meanMatrix
=

2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677
2.9690	2.9445	2.8361	2.4879	2.8540	1.6481	2.5677 ans

=

3.5000 3.5000 3.5000 3.5000 3.5000 3.5000 3.5000

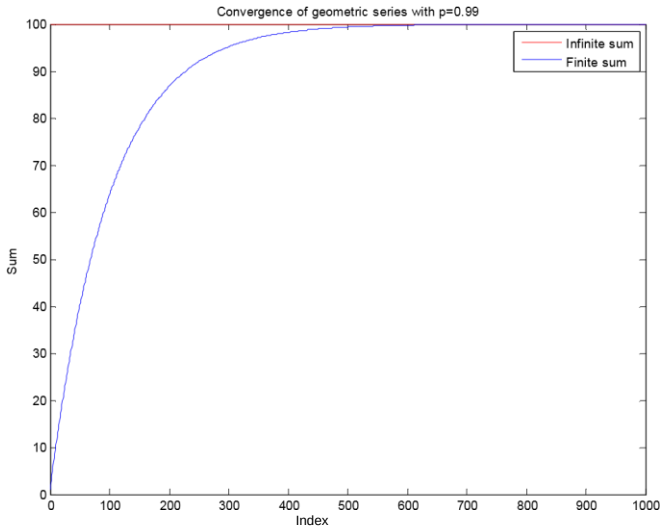
Grades: BCBBBACCBCCCCAB

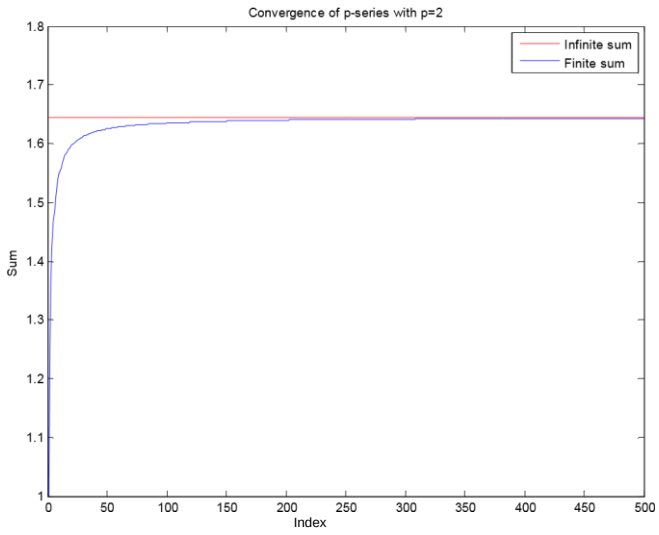
10. The seriesConvergence script is pasted below

```
% seriesConvergence
% this script plots two series to verify that they converge to the %
analytical value

% define the constants
p=0.99; k=0:1000;

% calculate each term in the series geomSeries=p.^k;

% calculate the infinite sum theoreticalValue=1/(1-p);

% plot theory and cumulative sum figure
```

```
plot([0 max(k)],theoreticalValue*ones(1,2),'r'); hold
on
plot(k,cumsum(geomSeries)); xlabel('Index'); ylabel('Sum');
title(['Convergence of geometric series with p=' num2str(p)]);
legend('Infinite sum','Finite sum');



% define the new constants
p=2; n=1:500;


% calculate each term in the p-series pSeries=(1./n).^p;


% calculated theoretical answer, which is the solution to the basel problem
baselValue=pi^2/6;


% plot theory and cumulative sum figure
plot([1 max(n)],baselValue*ones(1,2),'r'); hold
on
plot(n,cumsum(pSeries)); xlabel('Index');
ylabel('Sum'); title('Convergence of p-
series with p=2'); legend('Infinite
sum','Finite sum');
```

The figures generated by seriesConvergence are below





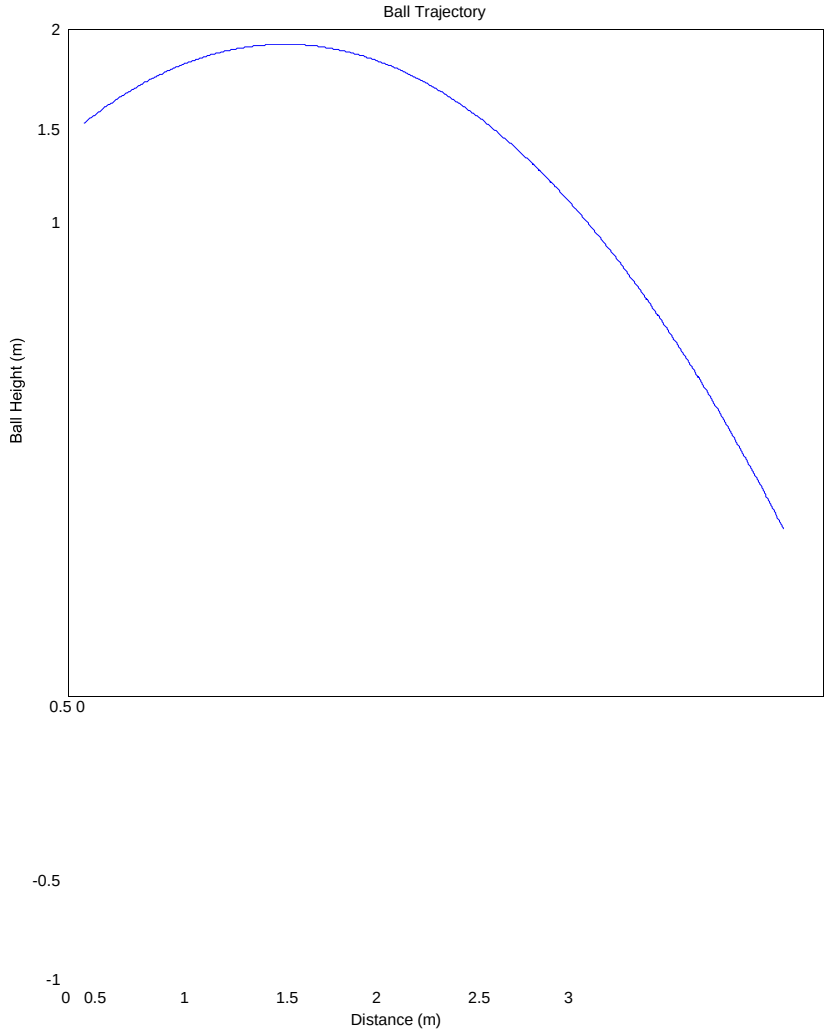
11. The throwBall script is pasted below

```
% throwBall
% this is a script that throws a ball of a set mass at a specified angle
% and with a specified initial velocity and calculates where the ball lands

% define the constants h=1.5; %meters
g=9.8; %gravitational acceleration in m/s^2
v=4; %m/s theta=45; % degrees

% make a time vector t=linspace(0,1,1000);

% calculate the x and y positions as a function of time
x=v*cos(theta/180*pi)*t; y=h+v*sin(theta/180*pi)*t-1/2*g*t.^2;

% find when it hits the ground
ind=find(y<0,1,'first'); distance=x(ind);
disp(['The ball hits the ground at a distance of ' num2str(distance) '
meters']);

% plot the ball's trajectory
figure plot(x,y)
xlabel('Distance (m)');
ylabel('Ball Height (m)');
title('Ball Trajectory');

% plot the ground hold
on
plot([0 max(x)],[0 0],'k--');
```

The output printed to the screen upon running throwBall is:

The ball hits the ground at a distance of 2.5821 meters

The throwBall script generates this figure:



12. The encrypt script is pasted below:

```
% encrypt
% this is a simple encryption script that takes a string and shuffles the
% letters around and then replaces the letters

% define the original string original='This
is my top secret message!';

% make the encoding indices vector encodeInds=randperm(length(original));

% apply the shuffled indices vector to the original
encoded=original(encodeInds);

% make the decode indices vector
temp=[encodeInds;1:length(original)]';
temp=sortrows(temp); decodeInds=temp(:,2);

% unshuffle the shuffled message decoded=encoded(decodeInds);

% display all three phrases disp(['Original:
' original]); disp(['Encoded : ' encoded]);
disp(['Decoded : ' decoded]);

% verify that the decoding happened correctly
correct=strcmp(original,decoded); disp(['Decoded correctly (1
true, 0 false): ' num2str(correct)]);
```

The screen output of encrypt.m is:

Original: This is my top secret message!

Encoded : iis !sgep etsasr Tmhsceomeyt

Decoded : This is my top secret message!

Decoded correctly (1 true, 0 false): 1