Engineering Assignment Coversheet

Please note that you:

- Must keep a full copy of your submission for this assignment
- Must staple this assignment
- Must NOT use binders or plastic folders except for large assignments

Student Number(s)

872301
931914

Group Code (if applicable):

~~5~~ 10

Assignment Title:	Matlab Workshop_1
Subject Number:	ELEN90054
Subject Name:	Probability and Random Models
Student Name:	Yue Chang, ZheHan Chen
Lecturer/Tutor:	Pasha Tolmachev
Due Date:	23/3/2018

For Late Assignments Only

Has an extension been granted? Yes / No (circle)

A per-day late penalty may apply if you submit this assignment after the due date/extension. Please check with your Department/coordinator for further information.

Plagiarism

Plagiarism is the act of representing as one's own original work the creative works of another, without appropriate acknowledgment of the author or source.

Collusion

Collusion is the presentation by a student of an assignment as his or her own which is in fact the result in whole or in part of unauthorised collaboration with another person or persons. Collusion involves the cooperation of two or more students in plagiarism or other forms of academic misconduct.

Both collusion and plagiarism can occur in group work. For examples of plagiarism, collusion and academic misconduct in group work please see the University's policy on Academic Honesty and Plagiarism: http://academichonesty.unimelb.edu.au/ •

Plagiarism and collusion constitute cheating. Disciplinary action will be taken against students who engage in plagiarism and collusion as outlined in University policy. Proven involvement in plagiarism or collusion may be recorded on my academic file in accordance with Statute 13.1.18.

STUDENT DECLARATION

Please sign below to indicate that you understand the following statements:

I declare that:

- This assignment is my own original work, except where I have appropriately cited the original source.
- This assignment has not previously been submitted for assessment in this or any other subject.

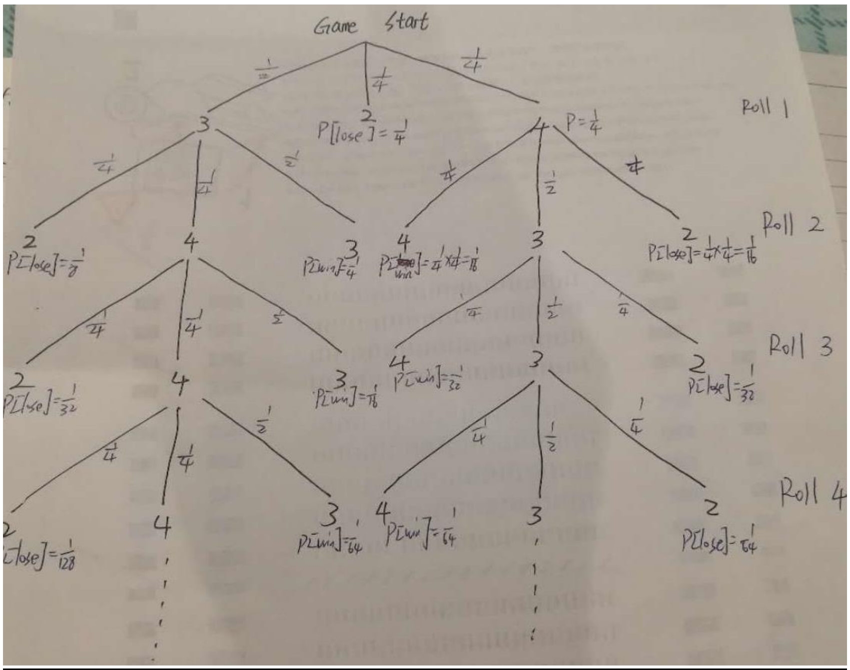
For the purposes of assessment, I give the assessor of this assignment the permission to:

- Reproduce this assignment and provide a copy to another member of staff; and
- Take steps to authenticate the assignment, including communicating a copy of this assignment to a checking service (which may retain a copy of the assignment on its database for future plagiarism checking).

Student signature .... Date 21/03/2018

Prelab

I-a. Draw a tree diagram of the system



I-b. Specify the sample space S for this game

There are two kinds of samples for this game：

1). S={2}：The player loses immediately (T(1) equals to 2)

2). S={x………..y}, where x∈[3,4]&y∈[2,4]; both x and y are integers
If y=x, the player wins the game
If y=2,the player loses the game

I-c. What are the possible outcomes? (total roll)

The possible outcomes are integers larger no less than 1:
Roll:[1,∞)

I-d. Show that the probabilities of the outcomes

I will calculate the probabilities of the outcomes following the sample space written in **I-b.(define k as the number of rolls)**

1). $P[\{2\}] = \frac{3}{6} * \frac{3}{6} = \frac{1}{4}$ (k=1)

2). $P[\{3 \dots \dots 3\}] = (\frac{1}{2})^2 * (\frac{1}{4})^{k-2}$ (k>=2)

3). $P[\{3 \dots \dots 2\}] = (\frac{1}{2}) * (\frac{1}{4})^{k-1}$ (k>=2)

4). $P[\{4 \dots\dots 4\}] = (\frac{1}{4})^2 * (\frac{1}{2})^{k-2}$ (k>=2)

5). $P[\{4 \dots\dots 2\}] = (\frac{1}{4})^2 * (\frac{1}{2})^{k-2}$ (k>=2)

Now I will prove that when k is close to infinity, the total probabilities of the outcomes will equals to 1:

$$P = P[\{2\}] + \sum_{k=2}^{\infty} (P[\{3 \dots\dots 3\}] + P[\{3 \dots\dots 2\}] + P[\{4 \dots\dots 4\}] + P[\{4 \dots\dots 2\}])$$

$$= \frac{1}{4} + \sum_{k=2}^{\infty} [(\frac{1}{2})^2 * (\frac{1}{4})^{k-2} + (\frac{1}{2}) * \left(\frac{1}{4}\right)^{k-1}] + \sum_{k=2}^{\infty} [\left(\frac{1}{4}\right)^2 * \left(\frac{1}{2}\right)^{k-2} + \left(\frac{1}{4}\right)^2 * \left(\frac{1}{2}\right)^{k-2}]$$

$$= \frac{1}{4} + \sum_{k=2}^{\infty} [\frac{3}{8} * \left(\frac{1}{4}\right)^{k-2}] + \sum_{k=2}^{\infty} [\frac{1}{8} * \left(\frac{1}{2}\right)^{k-2}]$$

$$= \frac{1}{4} + \sum_{k=2}^{\infty} [\frac{3}{8} * \left(\frac{1}{4}\right)^{k-2}] + \sum_{k=2}^{\infty} [\frac{1}{8} * \left(\frac{1}{2}\right)^{k-2}]$$

$$= \frac{1}{4} + \sum_{i=0}^{\infty} [\frac{3}{8} * \left(\frac{1}{4}\right)^{i}] + \sum_{i=0}^{\infty} [\frac{1}{8} * \left(\frac{1}{2}\right)^{i}] \quad (i = k - 2)$$

$$= \frac{1}{4} + \frac{\frac{3}{8}\left[1 - \left(\frac{1}{4}\right)^i\right]}{1 - \frac{1}{4}} + \frac{\frac{1}{8}\left[1 - \left(\frac{1}{2}\right)^i\right]}{1 - \frac{1}{2}} \quad hint: \left(\frac{1}{4}\right)^i = \left(\frac{1}{2}\right)^i = 0 \; when \; i = \infty$$

$$= \frac{1}{4} + \frac{1}{2} + \frac{1}{4} = 1$$

Proved

I-e.

The probability that the game never finishes is equals to 0.

From **I-d,** we know that when the number of rolls equals to infinity, which means k=∞, we have proved that the total probabilities of the five outcomes equals to 1. It means that there is no other conditions except these five outcomes. So the condition the game never finishes is not exist, which means the probability equals to 0

I-f.

From I-d, we know that there are two conditions the player will win the game:

$P[\{3 \dots\dots 3\}] = (\frac{1}{2})^2 * (\frac{1}{4})^{k-2}$ (k>=2)

$P[\{4 \dots\dots 4\}] = (\frac{1}{4})^2 * (\frac{1}{2})^{k-2}$ (k>=2)

The gamer cannot win the game in roll 1, so we can just add the probabilities of two conditions above to get the probability the play win the game.(k>=2)

$$P(\text{win}) = P[\{3 \dots \dots 3\}] + P[\{3 \dots \dots 3\}] = \sum_{k=2}^{\infty}[\left(\frac{1}{2}\right)^2 * \left(\frac{1}{4}\right)^{k-2}] + \sum_{k=2}^{\infty}[\left(\frac{1}{4}\right)^2 * \left(\frac{1}{2}\right)^{k-2}$$

$$= \sum_{i=0}^{\infty}[\left(\frac{1}{2}\right)^2 * \left(\frac{1}{4}\right)^{i}] + \sum_{i=0}^{\infty}[\left(\frac{1}{4}\right)^2 * \left(\frac{1}{2}\right)^{i}]$$

$$= \frac{\frac{1}{4}\left[1-\left(\frac{1}{4}\right)^i\right]}{1-\frac{1}{4}} + \frac{\frac{1}{16}\left[1-\left(\frac{1}{2}\right)^i\right]}{1-\frac{1}{2}}$$

$$= \frac{1}{3} + \frac{1}{8} = \frac{11}{24}$$

I-g. code is as follows:

```
clc;
clear all;
winexample = 0;
i = 1;
  while 1
     d1 = randi(2);
     d2 = randi(2);
     outcome(i) = d1+d2;       %The result(outcome) equals the total number of twp dices


     if ((outcome(1)~=2))
          winexample = outcome(1,1);%If first roll is not 2,save its number as winexample
     end

     if( outcome(i) == 2) % Whenever the outcome equals to 2(whatever times), the gamer loses the
game at once
          disp('player lost');
          z = sprintf('player has lost on NO.%d throw ',i);
          disp(z)
          break
     elseif((outcome(i) == winexample)&&(i~=1) )% After roll 1,if the gamer get the same outcome as
roll 1, he will win
          y = sprintf('The number of first roll is %d',winexample);
          x = sprintf('player has won on NO.%d throw ',i);
          disp(y)
          disp(x)
          break
     end
      i = i+1;
end
```

Output of code:

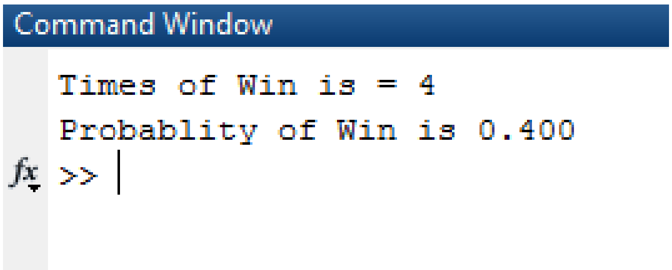
```
Command Window
The number of first roll is 4
player has won on NO.3 throw
>>
```

I-h. code is as follows:

```
clc;
clear all;
winexample = 0;
n=10;
wintimes = 0;
i = 1;
for j=1:1:n
    while 1
        d1 = randi(2);
        d2 = randi(2);
        outcome(i) = d1+d2;          %The result(outcome) equals the total number of twp dices


    if ((outcome(i) == 2)) % Whenever the outcome equals to 2(whatever times), the gamer loses the game at once


            break
        end

        if ((outcome(1)~=2))
            winexample = outcome(1);     %If first roll is not 2,save its number as winexample
        end

        if((outcome(i) == winexample)&&(i~=1) )     % After roll 1,if the gamer get the same outcome as roll 1, he will win
            wintimes = wintimes+1;
            break
        end
        i = i+1;
```

```
        end
        i=1;
end
p = wintimes/n; % Probability =wintimes divided by total times n
x = sprintf('Times of Win is = %d ',wintimes);
y = sprintf('Probablity of Win is %.3f ',p);
disp(x)
disp(y)
```

Output of code:

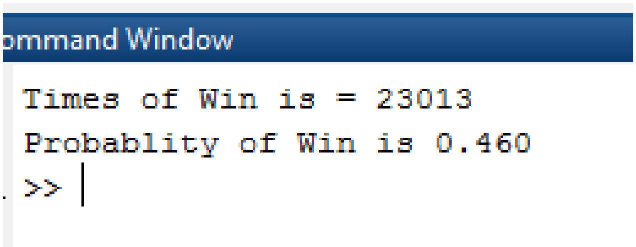


I-i. code is as follows:

```
clc;
clear all;
winexample = 0;
n=50000;
wintimes = 0;
i = 1;
for j=1:1:n
        while 1
                d1 = randi(2);
                d2 = randi(2);
                outcome(i) = d1+d2;            %The result(outcome) equals the total number of twp dices


        if ((outcome(i) == 2)) % Whenever the outcome equals to 2(whatever times), the gamer loses the
game at once


                        break
                end

                if ((outcome(1)~=2))
                        winexample = outcome(1);     %If first roll is not 2,save its number as winexample
```

```
            end

            if((outcome(i) == winexample)&&(i~=1) )    % After roll 1,if the gamer get the same
outcome as roll 1, he will win
                wintimes = wintimes+1;
                break
            end
            i = i+1;

        end
        i=1;
end
p = wintimes/n; % Probability =wintimes divided by total times n
x = sprintf('Times of Win is = %d ',wintimes);
y = sprintf('Probablity of Win is %.3f ',p);
disp(x)
disp(y)
```

Output of code:



I-J. Comment:

We can see from I-h and I-i that when the number of experiments is 10, P[win]=0.4 and the probabilities changes when we run the code again. So the result of I-h related to my answer of I-f poorly because the number of experiments is not larger enough. Then in I-i, we increase the times of experiments to 50000,which is absolutely larger enough, and then P[win]=0.460 approximately equals to the result in I-f (11/24), and remain almost constant when we run the code again. Thus the result in I-i related to my answer of I-f very well.

1) code is as follows:

```
clc;
clear all;
winexample = 0;
i = 1;
  while 1
    d1 = randi(6);
```

```
        d2 = randi(6);
        outcome(i) = d1+d2;       %The result(outcome) equals the total number of twp dices


        if ((outcome(1)~=2))
            winexample = outcome(1,1);%If first roll is not 2,save its number as winexample
        end

        if( outcome(i) == 2) % Whenever the outcome equals to 2(whatever times), the gamer loses the game at once
            disp('player lost');
            z = sprintf('player has lost on NO.%d throw ',i);
            disp(z)
            break

        elseif((outcome(1,i) == winexample)&&(i~=1) )% After roll 1,if the gamer get the same outcome as roll 1, he will win
            y = sprintf('The number of first roll is %d',winexample);
            x = sprintf('player has won on NO.%d throw ',i);
            disp(y)
            disp(x)
            break
        end
         i = i+1;
end
```

Output of code:

```
Command Window
The number of first roll is 4
player has won on NO.5 throw
>>
```

2) code is as follows:

```
clc;
clear all;
winexample = 0;
n=10;
wintimes = 0;
i = 1;
for j=1:1:n
    while 1
```

```
        d1 = randi(6);
        d2 = randi(6);
        outcome(i) = d1+d2;          %The result(outcome) equals the total number of twp dices


    if ((outcome(i) == 2)) % Whenever the outcome equals to 2(whatever times), the gamer loses the game at once


            break
        end

        if ((outcome(1)~=2))
            winexample = outcome(1);    %If first roll is not 2,save its number as winexample
        end

        if((outcome(i) == winexample)&&(i~=1) )    % After roll 1,if the gamer get the same outcome as roll 1, he will win
            wintimes = wintimes+1;
            break
        end
        i = i+1;

    end
    i=1;
end
p = wintimes/n; % Probability =wintimes divided by total times n
x = sprintf('Times of Win is = %d ',wintimes);
y = sprintf('Probablity of Win is %.3f ',p);
disp(x)
disp(y)
```

Output of code:

```
Command Window
  Times of Win is = 8
  Probablity of Win is 0.800
>>
```

3) code is as follows:

```
clc;
clear all;
winexample = 0;
n=50000;
wintimes = 0;
i = 1;
for j=1:1:n
    while 1
        d1 = randi(6);
        d2 = randi(6);
        outcome(i) = d1+d2;          %The result(outcome) equals the total number of twp dices


    if ((outcome(i) == 2)) % Whenever the outcome equals to 2(whatever times), the gamer loses the
game at once


            break
        end

        if ((outcome(1)~=2))
            winexample = outcome(1);    %If first roll is not 2,save its number as winexample
        end

        if((outcome(i) == winexample)&&(i~=1) )    % After roll 1,if the gamer get the same
outcome as roll 1, he will win
            wintimes = wintimes+1;
            break
        end
        i = i+1;

    end
    i=1;
end
p = wintimes/n; % Probability =wintimes divided by total times n
x = sprintf('Times of Win is = %d ',wintimes);
y = sprintf('Probablity of Win is %.3f ',p);
disp(x)
disp(y)
```

Output of code:

```
Command Window
Times of Win is = 38298
Probablity of Win is 0.766
fx >>
```

Result: From the comment in I-j, we know that if we want to estimate the probability the play wins, we should use the result when the number of experiments is 50000,which equals to 0.766.
Thus P[win]=0.766

II-a． Code is as follows

```
clc;
clear all;
car = randi(3);                  %give a random number to represent the car door
goat = car;                      %give the goat door which host will show an intial value
select = randi(3);               %give a random number to represent the player selected door
x = sprintf('door player select is %d',select);
disp(x)
%select from the three number for the host to show the goat door%
%the number shouldn't be car door or the player selected door%
while((goat == select)||(goat == car))
    goat = randi(3);
end
y = sprintf('show goat is %d',goat);
disp(y)
monty = goat;                    %give the door left for host a intial value%
%the door left for host shouldn't be the already shown goat door or the player's selected door%
while((monty == goat)||(monty == select))
    monty = randi(3);
end
%procedure 1: not switch%
%if the play's selected door is car%
if(select == car)
    %so he not switch he win%
    disp('procedure 1:not switch player win')
%and else he doesn't choose the car door%
else
    %and he does't switch he lose%
```

```
        disp('procedure 1:not switch player lose')
end
%procedure 2: switch%
%if the door left for monty is car%
if(monty == car)
        %so the player switch he win%
      disp('procedure 2:switch player win')
%and else the door left for monty is not a car door%
else
        %so player switch he lose%
        disp('procedure 2:switch player lose')
end
```

Output of code:

```
door player select is 1
show goat is 2
procedure 1:not switch player lose
procedure 2:switch player win
```

II-b . Code is as follows:

```
clc;
clear all;
n=10;
swin = 0;
nwin = 0;
%repeat IIa for n times
for j=1:1:n
        car = randi(3);
        goat = car;
        select = randi(3);
        while((goat == select)||(goat == car))
                goat = randi(3);
        end
        monty = goat;
        while((monty == goat)||(monty == select))
                monty = randi(3);
        end
        if(select == car)
                disp('procedure 1:not switch player win')
                nwin = nwin + 1;
        else
```

```
            disp('procedure 1:not switch player lose')
        end
        if(monty == car)
            disp('procedure 2:switch player win')
            swin = swin + 1;
        else
            disp('procedure 2:switch player lose')
        end
end
%display the total number of procedure 1 when the player not switch win%
x = sprintf('Total number of procedure 1 NOT Switch Win = %d ',nwin);
disp(x)
%display the total number of procedure 2 when the player switch win%
x = sprintf('Total number of procedure 2 Switch Win = %d ',swin);
disp(x)
p = swin/n;
x = sprintf('Probability of Switch Win = %.3f ',p);
disp(x)
```

Output of code:

```
procedure 1:not switch player lose
procedure 2:switch player win
procedure 1:not switch player lose
procedure 2:switch player win
procedure 1:not switch player win
procedure 2:switch player lose
procedure 1:not switch player lose
procedure 2:switch player win
procedure 1:not switch player lose
procedure 2:switch player win
procedure 1:not switch player lose
procedure 2:switch player win
procedure 1:not switch player win
procedure 2:switch player lose
procedure 1:not switch player win
procedure 2:switch player lose
procedure 1:not switch player lose
procedure 2:switch player win
procedure 1:not switch player lose
procedure 2:switch player win
Total number of procedure 1 NOT Switch Win = 3
Total number of procedure 2 Switch Win = 7
Probability of Switch Win = 0.700
>>
```

II-c．Code is as follows:

```
%change IIb's code n to 50000
clc;
clear all;
n=50000;
swin = 0;
nwin = 0;
for j=1:1:n
    car = randi(3);
    goat = car;
    select = randi(3);
    while((goat == select)||(goat == car))
        goat = randi(3);
    end
    monty = goat;
    while((monty == goat)||(monty == select))
        monty = randi(3);
    end
    if(select == car)

        nwin = nwin + 1;
    else
    end
    if(monty == car)

        swin = swin + 1;
    else
    end
end
x = sprintf('Total number of Switch Win = %d ',swin);
disp(x)
p = swin/n;
x = sprintf('Probability of Switch Win = %.3f ',p);
disp(x)
x = sprintf('Total number of NOT Switch Win = %d ',nwin);
disp(x)
```

Output of code:

```
Total number of procedure 1 NOT Switch Win = 16674
Total number of procedure 2 Switch Win = 33326
Probability of Switch Win = 0.667
```

II-d

Because ‘switch win’ and ‘switch lose’ are complementary events, and ‘not switch win’ and ‘switch lose’ are the same events. So we have:

P(not switch win) = P(switch lose) = 1-P(switch win) = 1/3

So, P(switch win) = 2/3

II-e

From II-b result we can see that the probability of the player switch and win is 0.7, and from II-c result the probability of the player switch and win is 0.667. Both of the results are approximately equals to 2/3, so we can say the simulations are successful. And II-c result is closer to 2/3, we can say that with more experiment times, the result is more accurate.

II-f

From the simulation and calculation above we can conclude that the player switch and win has a higher probability, so if the player wants to win a car, he should switch.