In [1]:
```
### Question 1

#A robotics company designs legged robots and wheeled robots for agricul
tural applications. They have a user interface that allows potential use
rs to enter details of their requirements. You have to develop an object
 oriented program to keep a track of these user requirements, inform cus
tmers what can be made and what cannot be made, and provide an estimate
 of costs involved.

### Question 1.1
#Develop a class called "robot" that takes the following user informatio
n:
#* Name of the user
#* Area of the farm in $m^2$
#* What is grown in the farm
#* Terrain conditions (flat, hilly)
#* Expected maximum acceleration

class robot:
    num = 0
    def __init__(self):
        robot.num += 1
        self.UserNum = robot.num
        #Name of the user
        self.name = input("Please type your name: ")
        #Area of the farm
        self.area = float(input("What is the estimated area of your farm
 in square meters? "))
        #What is grown in the farm
        self.farmtype = input("What is the farm about? ")
        #Terrain conditions (flat, hilly)
        self.terrain = input("Is your farm terrain hilly or flat? ")
        #Expected maximum acceleration
        self.maxacc = float(input("What is the expected maximum accelera
tion of the robot? "))

myrobot = robot()
```

```
Please type your name: Thrish
What is the estimated area of your farm in square meters? 3450
What is the farm about? Wheat
Is your farm terrain hilly or flat? flat
What is the expected maximum acceleration of the robot? 34
```

In [7]:
```
### Question 1.2
#Extend the above class to inform the user whether the robot can be desi
gned or not
#based on the maximum expected acceleration, assuming that the company c
an make robots
#that can move at a maximum acceleration of $10 m/sec^2$ if the terrain
 is hilly, $20 m/sec^2$ if it is flat.
class robot:
    num = 0
    def __init__(self):
        robot.num += 1
        self.UserNum = robot.num
        #Name of the user
        self.name = input("Please type your name: ")
        #Area of the farm
        self.area = float(input("What is the estimated area of your farm
 in square meters? "))
        #What is grown in the farm
        self.farmtype = input("What is the farm about? ")
        #Terrain conditions (flat, hilly)
        self.terrain = input("Is your farm terrain hilly or flat? ")
        #Expected maximum acceleration
        self.maxacc = float(input("What is the expected maximum accelera
tion of the robot? "))

    def testaccel(self):
        if self.maxacc > 20 and (self.terrain == 'flat' or self.terrain
== 'Flat'):
            print("Sorry your expected acceleration is beyond the 10m/se
c^2 we make for flat terrains")
        elif self.maxacc > 10 and (self.terrain == 'hilly' or self.terra
in == 'Hilly'):
            print("Sorry your expected acceleration is beyond 20m/sec^2
 we make for hilly terrains")
        else:
            print("Glad that your expected acceleration is within our ra
nge")

myrobot = robot()
myrobot.testaccel()
```

```
Please type your name: Thrish
What is the estimated area of your farm in square meters? 2300
What is the farm about? Wheat
Is your farm terrain hilly or flat? flat
What is the expected maximum acceleration of the robot? 15
Glad that your expected acceleration is within our range
```

In [14]:
```
### Question 1.3
#Assuming the user's acceleration requirement is within the company's ca
pability,
#extend the above class with a method to calculate and print the fuel re
quirement to
#cover the farm land to 2-decimal places based on the following informat
ion.

#Let the farm area be $a$.
#Annual fuel cost = $0.002 a^2 + 0.4 log(a)$ if the terrain condition is
 hilly.
#Annual fuel cost = $0.001 a^2 + 0.4 a$ if the terrain condition is fla
t.

from math import *

class robot:
    num = 0
    def __init__(self):
        robot.num += 1
        self.UserNum = robot.num
        #Name of the user
        self.name = input("Please type your name: ")
        #Area of the farm
        self.area = float(input("What is the estimated area of your farm
 in square meters? "))
        #What is grown in the farm
        self.farmtype = input("What is the farm about? ")
        #Terrain conditions (flat, hilly)
        self.terrain = input("Is your farm terrain hilly or flat? ")
        #Expected maximum acceleration
        self.maxacc = float(input("What is the expected maximum accelera
tion of the robot? "))

    def testaccel(self):
        if self.maxacc > 20 and (self.terrain == 'flat' or self.terrain
== 'Flat'):
            print("Sorry your expected acceleration is beyond the 10m/se
c^2 we make for flat terrains")
        elif self.maxacc > 10 and (self.terrain == 'hilly' or self.terra
in == 'Hilly'):
            print("Sorry your expected acceleration is beyond 20m/sec^2
 we make for hilly terrains")
        else:
            print("Glad that your expected acceleration is within our ra
nge")
    def fuel(self):
        if self.maxacc <= 20 and (self.terrain == 'flat' or self.terrain
 == 'Flat'):
            a = self.area
            f = 0.0001*a**2 + 0.4*a
            #print("The expected fuel cost is £{}" .format(f))
        elif self.maxacc <= 10 and (self.terrain == 'hilly' or self.terr
ain == 'Hilly'):
            a = self.area
            f = 0.0002*a**2 + 0.4*log(a)
```

```
            #print("The expected fuel cost is £{}" .format(f))
        else:
            print("The expected acceleration is too much!")
        return f

myrobot = robot()
myrobot.testaccel()
ff = myrobot.fuel()
print("The expected annual fuel cost is £{:.2f}" .format(ff))
```

```
Please type your name: Thrish
What is the estimated area of your farm in square meters? 2300
What is the farm about? Wheat
Is your farm terrain hilly or flat? flat
What is the expected maximum acceleration of the robot? 15
Glad that your expected acceleration is within our range
The expected annual fuel cost is £1449.00
```

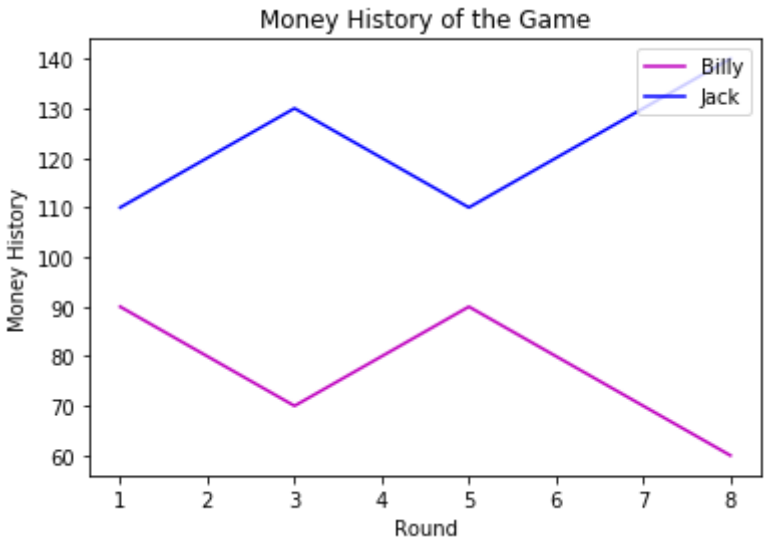
In [19]:
```
class wheeledrobot(robot):
    def __init__(self):
        robot.__init__(self)
        self.mass = float(input("What is the payload you expect to carry
 on the robot: "))
        if self.mass > 30 and (self.terrain == 'hilly' or self.terrain =
= 'Hilly'):
            print("Our wheeled robots are not suitable to carry payloads
 greater than 30kgs in hilly terrains")
        if self.mass > 50 and (self.terrain == 'flat' or self.terrain ==
 'Flat'):
            print("Our wheeled robots are not suitable to carry payloads
 greater than 50kgs in flat terrains")
        else:
            tor = (40 + self.mass)*self.maxacc
            print("The expected peak torque is {}Nm" .format(tor))

myrob = wheeledrobot()
```

```
Please type your name: Thrish
What is the estimated area of your farm in square meters? 2300
What is the farm about? Wheat
Is your farm terrain hilly or flat? hilly
What is the expected maximum acceleration of the robot? 7
What is the payload you expect to carry on the robot: 20
The expected peak torque is 420.0
```

In [1]:
```
## Question 3

#Two players Jack and Billy plays a dice game. They will both start the
 game with 100 GBP.
#In each round each player will roll 3 dice each having number between 1
-6. The person who has the higher score
#(add the number of the three dice) wins. The winner gets 10 GBP from th
e other player.
#If they have the same score, they need to both roll the dice again.

## Question 3.1
#Write a class called **player** that has methods to roll 3-dice, to add
 £10 if one wins and to substract £10 if one loses.

## Question 3.2
#Create instances for Jack and Billy. Do the following after 8-rounds of
 playing.

#* print how much money each of them have at the end of the game.
#* print how many rounds each of them wins.
#* plot the total amount of money each player had at each round.
#Label x and y axes, add a title for the plot, and show legends for both
 players.

import random
import matplotlib.pyplot as plt

class player:
 def __init__ (self,name,money):
  self.name = name
  self.money = money
  self.scorehistory = []
  self.moneyhistory = []
  self.numwin = 0
  self.numlose = 0

 def getscorehistory():
  return self.scorehistory

 def win(self):
  self.money += 10
  self.moneyhistory.append(self.money)
  self.numwin += 1
  return self.money

 def lose(self):
  self.money -= 10
  self.moneyhistory.append(self.money)
  self.numlose += 1
  return self.money

 def rolldice(self):
     score = 0
     for x in range(3):
         score += random.randint(1,7)
     self.scorehistory.append(score)
```

```
     return score
playerONE = player("Billy",100)
playerTWO = player("Jack",100)

Round = 8
history = list(range(1,Round+1))
for _ in range(Round):
    score1 = playerONE.rolldice()
    score2 = playerTWO.rolldice()
    while (score1 == score2):
        score1 = playerONE.rolldice()
        score2 = playerTWO.rolldice()
    if score1 > score2:
        playerONE.win()
        playerTWO.lose()
    elif score1 < score2:
        playerTWO.win()
        playerONE.lose()

print('########Game Begin#########')
print('{} and {} joined the game. They both started the game with 100 GB
P'.format(playerONE.name,playerTWO.name))
print('{} has {} GBP at the end of the game'.format(playerONE.name,playe
rONE.money))
print('{} has {} GBP at the end of the game'.format(playerTWO.name,playe
rTWO.money))
print('The score history of {} is {}'.format(playerONE.name,playerONE.sc
orehistory))
print('The score history of {} is {}'.format(playerTWO.name,playerTWO.sc
orehistory))
fig, ax = plt.subplots()
ax.plot(history,playerONE.moneyhistory,'m-',label=playerONE.name)
ax.plot(history,playerTWO.moneyhistory,'b-',label=playerTWO.name)
plt.xlabel("Round")
plt.ylabel("Money History")
plt.title("Money History of the Game")
legend = ax.legend(loc='upper right', shadow=False)
plt.show()
print('The Money history of {} is {}'.format(playerONE.name,playerONE.mo
neyhistory))
print('The Money history of {} is {}'.format(playerTWO.name,playerTWO.mo
neyhistory))
print('{} won {} times'.format(playerONE.name,playerONE.numwin))
print('{} won {} times'.format(playerTWO.name,playerTWO.numwin))
```

```
########Game Begin#########
Billy and Jack joined the game. They both started the game with 100 GBP
Billy has 60 GBP at the end of the game
Jack has 140 GBP at the end of the game
The score history of Billy is [11, 10, 10, 14, 12, 7, 13, 12, 11]
The score history of Jack is [13, 13, 14, 12, 9, 15, 13, 13, 15]
```




```
The Money history of Billy is [90, 80, 70, 80, 90, 80, 70, 60]
The Money history of Jack is [110, 120, 130, 120, 110, 120, 130, 140]
Billy won 2 times
Jack won 6 times
```