

```
#import required modules
import math
import random
```

```
#Functions are below:
```

```
#setgame defines variables from user input. P = # players, G = total # gelt, S = # gelt each gets
to start with,
```

```
# D = each spin of the dreidel, T = Turn number
```

```
def setgame():
    setgame = 1
    while setgame == 1:
        print("Welcome to Dreidel!")
        P = int(raw_input("How many players are in this game?"))
        print(P)
        G = int(raw_input("How much gelt do you have to play with?"))
        print(G)
        if G < P:
            print("You need at least as much gelt as people playing!")
        else:
            setgame = 0
```

```
# the number of gelt each person will get to start with
```

```
start = math.floor(float(G)/(float(P)+1.))
```

```
#Create the list (L) to track how much gelt each player has and distributes the gelt.
```

```
#Index 0 is the pot. All other indices are that player.
```

```
#A is the number of players who are still "In"
```

```
#If players evenly divides gelt, pot gets no gelt.
```

```
if start == 0:
```

```
    L = [1] * P
```

```
    L.insert(0, 0)
```

```
#Otherwise players all get start gelt and pot gets remainder
```

```
else:
```

```
    L = [0] * P
```

```
    for i in range(P):
```

```
        L[i] = start
```

```
        if start * P < G:
```

```
            L.insert(0, G - (start * P))
```

```

        else:
            L.insert(0, start)
    print("The pot has", L[0], "gelt and player(s) have ", L[1], "gelt \n")
    print("The dreidel falls as follows: (1=nun, 2=gimmel, 3=hey, 4=shin) \n")

# 1=nun, 2=gimmel, 3=hey, 4=shin
return(P,G,L)

#FOR LATER TO SUPPRESS OUTPUT, add oflag to function. Every print will become
# if oflag: and then print. This way output will be suppressed when oflag (output flag)
# is false, when I want to graph things.
#
def Turn(P,Llocal,T,oflag):
    #This function takes the number of players, the array with the record of each players' gelt
    # and the turn number as arguments, and returns them again after it simulates one round of
    play
    # (ie, each player spins one time and gelt are re-assigned accordingly.)

    #The player variable keeps track of which player is spinning
    player=1
    while player < P+1:
        #If a player has no gelt, they do not get to roll. This might be changed later.
        if Llocal[player] <= 0:
            Llocal[player] = 0
            pass
        elif Llocal[player] > 0:
            #This is where the turns are counted.
            T=T+1
            #This is the dreidel spin.
            D = random.randint(1,4)
            if D == 1:
                D="nun"
            #Print results of this roll.
            if oflag:
                pass
            else:
                print("Player ", player, "spun a ", D)
                print("There are ", P, "Players remaining")
                print('[pot, players]')
                print(Llocal)
                print()

```

```

pass
#nothing happens
elif D == 2:
#SOMETHING IS WRONG WITH GIMMEL AGAIN! With L[player]
D="gimmel"
if oflag:
pass
else:
print("Player ", player, "spun a ", D)
#player gets L[0] gelt and all players put back one gelt.
d = Llocal[player] + Llocal[0]
Llocal[player] = d
Llocal[0] = P
k=Llocal[0]
#Check if putting back one caused any player to lose, update P.
for losscheck in range(1,len(Llocal)):
if Llocal[losscheck] > 0:
    Llocal[losscheck] = Llocal[losscheck]-1
    if Llocal[losscheck]==0:
        P=P-1
        if oflag:
            pass
        else:
            print("Player ", losscheck, "is out.")
            #Print results of this roll.
            print("There are ", P, "players remaining")
            print('[pot, players]')
            print(Llocal)
            print()
Llocal[0]=k
elif D == 3:
D="hey"
#Player takes half (rounded up) of L[0]
if oflag:
pass
else:
print("Player ", player, "spun a ", D)
d = Llocal[player] + math.ceil(float(Llocal[0])/2.)
Llocal[player] = d
#Check if player is out
if Llocal[player]==0:
P=P-1
if oflag:

```

```

        pass
    else:
        print("Player ", player, "is out.")
        Llocal[0] = math.floor(float(L[0])/2.)
        #Print results of this roll.
        if oflag:
            pass
        else:
            print("There are ", P, "players remaining")
            print('[pot, players]')
            print(Llocal)
            print()
            elif D == 4:
                D="shin"
            if oflag:
                pass
            else:
                print("Player ", player, "spun a ", D)
                #Player puts 1 into L[0]
                d = Llocal[player] - 1
                Llocal[player] = d
                #Check if player is out
                if Llocal[player]==0:
                    P=P-1
                if oflag:
                    pass
                else:
                    print("Player ", player, "is out.")
                    Llocal[0] = Llocal[0] + 1
                    #Print results of this roll.
                    if oflag:
                        pass
                    else:
                        print("There are ", P, "players remaining")
                        print('[pot, players]')
                        print(Llocal)
                        print()
                        player = player+1

return(P,Llocal,T,oflag)

```

```
#Set round and turn numbers to 0. R=rounds, T=turns.
```

```
R=0
```

```
T=0
```

```
turns = 0
```

```
rounds = 0
```

```
#This puts the program in a loop so players can play again after each game.
```

```
reset= 1
```

```
while reset == 1:
```

```
    #setgame sets essential variables such as player number, gelt number, and gelt per player.
```

```
        P,G,L = setgame()
```

```
        Pinitial = P
```

```
        Linitial = L
```

```
    #Actual gameplay.
```

```
    output = raw_input("Supress text? (Y or N)")
```

```
    if output == 'Y':
```

```
        oflag=1
```

```
    else:
```

```
        oflag=0
```

```
##SOMETHING IS WRONG HERE, On hernando, won't let turns or rounds be initialized like this, sadface.
```

```
    gameplay = int(raw_input("How many games should I play?"))
```

```
    turns = [0] * gameplay
```

```
    rounds = [0] * gameplay
```

```
    for iteration in range(gameplay):
```

```
        P = Pinitial
```

```
        L = Linitial
```

```
        for j in range(P+1):
```

```
            #If pot gets all the gelt
```

```
            if L[0]==G:
```

```
                print("The pot won :(")
```

```
            #If there is one player (ie someone won) go to display who won
```

```
            elif P==1:
```

```
                break
```

```
            #If no one won, increment rounds and play another round.
```

```
            else:
```

```
                while P > 1:
```

```
                    R = R+1
```

```
                    P,L,T,oflag = Turn(P,L,T,oflag)
```

```
#Sometimes the pot has the most gelt even when someone won. This first if
#handles that scenario.
list.pop(L, 0)
if oflag:
    pass
else:
    print("Player ", L.index(max(L))+1, " Won!")
    print("The game lasted ", R, "rounds and ", T, "turns")

turns[iteration] = T
rounds[iteration] = R

print(T, "turns")
print(R, "rounds")
print(turns)
print(rounds)

reset = raw_input("Play again? 1 for yes 0 for no")
reset = int(reset)
```