```python
#Functions Challenge 35:  Loan Calculator App
from matplotlib import pyplot

def get_loan_info():
    """Get the basic information of a loan and store it in a dictionary"""
    #Create a blank dict to represent a loan
    loan = {}

    #Get user input for the categories of the loan
    loan['principal'] = float(input("Enter the loan amount: "))
    loan['rate'] = float(input("Enter the interest rate: "))/100
    loan['monthly payment'] = float(input("Enter the desired monthly payment amount: "))
    loan['money paid'] = 0

    return loan


def show_loan_info(loan, number):
    """Display the current loan status"""
    print("\n----Loan information after " + str(number) + " months----")
    for key, value in loan.items():
        print(key.title() + ": " + str(value))


def collect_interest(loan):
    """Update loan for interest per month"""
    #Divide by 12 to simulate collecting interest monthly
    loan['principal'] = loan['principal'] + loan['principal']*loan['rate']/12


def make_monthly_payment(loan):
    """Simulate making a monthly payment to pay down the principal"""
    loan['principal'] = loan['principal'] - loan['monthly payment']

    #You are required to make a full payment this month, you have not yet payed off your loan
    if loan['principal'] > 0:
        loan['money paid'] += loan['monthly payment']
    #You are not required to make a full payment this month, you have payed off your loan
    else:
        #For this else block, loan['principal'] will be negative
        loan['money paid'] += loan['monthly payment'] + loan['principal']
        loan['principal'] = 0


def summarize_loan(loan, number, initial_principal):
    """Display the results of paying off the loan"""
    print("\nCongraulations!  You paid off your loan in " + str(number) + " months!")
    print("Your initial loan was $" + str(initial_principal) + " at a rate of " + str(100*loan['rate']) + "%.")
    print("Your monthly payment was $" + str(loan['monthly payment']) + ".")
    print("You spent $" + str(round(loan['money paid'], 2)) + " in total.")

    #Calculate money spent on interest
    interest = round(loan['money paid'] - initial_principal, 2)
    print("You spent $" + str(interest) + " on interest!")


def create_graph(data, loan):
    """Create a graph to show the relationship between principal and time"""
    x_values = []   #These represent month numbers
```

```
60      y_values = []   #These represent corresponding principal values
61
62      #Loop through data set.  Point is a tuple.
63      #point[0] represents a month number, point[1] represents a principal value.
64      for point in data:
65          x_values.append(point[0])
66          y_values.append(point[1])
67
68      #Create a plot for x_values and y_values (month number and principal)
69      pyplot.plot(x_values, y_values)
70      pyplot.title(str(100*loan['rate']) + "% Interest With $" + str(loan['monthly
    payment']) + " Monthly Payment")
71      pyplot.xlabel("Month Number")
72      pyplot.ylabel("Principal of Loan")
73
74      #Display the created graph
75      pyplot.show()
76
77
78  #The main code
79  print("Welcome to the Loan Calculator App\n")
80
81  #Initialize variables
82  month_number = 0
83  my_loan = get_loan_info()
84  starting_principal = my_loan['principal']
85  data_to_plot = []
86
87  #Display starting conditions of loan
88  show_loan_info(my_loan, month_number)
89  input("Press 'Enter' to begin paying off your loan.")
90
91  #Simulate paying off the loan as long as the loan's principal > 0
92  while my_loan['principal'] > 0:
93      #You cannot get ahead of the interest, you will never pay off the loan so
    break
94      if my_loan['principal'] > starting_principal:
95          break
96
97      #It is possible to pay off the loan, so simulate a single month
98      #Increment month number, collect interest, make a payment, add data to plot,
    and show loan info
99      month_number += 1
100     collect_interest(my_loan)
101     make_monthly_payment(my_loan)
102     data_to_plot.append((month_number, my_loan['principal']))
103     show_loan_info(my_loan, month_number)
104
105 #The loan is either paid off in full or it can NEVER be paid off...
106 #The loan was paid off in full
107 if my_loan['principal'] <= 0:
108     summarize_loan(my_loan, month_number, starting_principal)
109     create_graph(data_to_plot, my_loan)
110 #The loan can NEVER be paid off, can't get ahead of interest
111 else:
112     print("\nYou will NEVER pay off your loan!!!")
113     print("You cannot get ahead of the interest! :-(")
```