IMPORTING REQUIRED LIBRARIES FOR ALGORITHM

#LIBRARIES  
import pandas as pd  
import os  
#Color Text  
import colorama  
from colorama import Back, Fore  
#Progress Bars  
from alive\_progress import alive\_bar  
import time  
#Delay output  
from time import sleep  
import csv  
#importing made-modules  
import Calculations.scoring as cal   
import Calculations.classification as classi  
import Displaytool.tools as tool  
from csv import DictWriter

ALGORITHM HEADER DISPLAY

#######-----------HEADER---------------########  
colorama.init(autoreset=True)  
print(Back.BLACK +"|--------------------WELCOME TO--------------------|")  
print(Back.BLACK +f"|-----------------------{Fore.RED}C{Fore.BLUE}S{Fore.CYAN}P{Fore.GREEN}A-{Fore.WHITE}----------------------|")  
print(Back.BLACK +"|---------------------ALGORITHM--------------------|")  
print(Back.BLACK +"|--------------------------------------------------|")  
print(Back.BLACK +f"|{Fore.GREEN}By: CS Warriors{Fore.WHITE}-----------------------------------|")  
print("\n")  
#######-----------END OF HEADER---------------########

|--------------------WELCOME TO--------------------|  
|-----------------------CSPA-----------------------|  
|---------------------ALGORITHM--------------------|  
|--------------------------------------------------|  
|By: CS Warriors-----------------------------------|

DISPLAYING LOADING ALIVE-PROGRESS BARS

#######-----------LOADING PROGRESS BAR--------########  
#A. LOADING BAR  
print("Loading Program :\n")  
with alive\_bar(100) as bar:  
 for i in range(100):  
 time.sleep(.050)  
 if i == 30:  
 print(f"{Fore.GREEN}[30%]{Fore.WHITE}Checking Libraries..")  
 if i == 50:  
 print(f"{Fore.GREEN}[50%]{Fore.WHITE}Checking Resources..")  
 if i == 75:  
 print(f"{Fore.GREEN}[75%]{Fore.WHITE}Optimizing..")  
 if i == 99:  
 print(f"{Fore.GREEN}[100%]{Fore.WHITE}Program Successfully Loaded")  
 bar()

Loading Program :  
  
on 30: [30%]Checking Libraries..  
on 50: [50%]Checking Resources..  
on 75: [75%]Optimizing..  
on 99: [100%]Program Successfully Loaded  
|████████████████████████████████████████| 100/100 [100%] in 5.1s (19.50/s)

IMPORTING DATASET AND CALL MODULES FOR CALCULATIONS AND CLASSIFICATION

#B. IMPORTING DATASET LOADING BAR  
print("\nImporting Dataset :\n")  
with alive\_bar(100) as bar:  
 for x in range(100):  
 time.sleep(.090)  
 if x == 25:  
 print(f"{Fore.GREEN}[25%]{Fore.WHITE}Checking Dataset..")  
 # Checking  
 dataset = pd.read\_csv('data/Origin COOP.csv')  
 #To display values in 2 decimal places  
 pd.options.display.float\_format = '{:.2f}'.format  
  
 if x == 50:  
 print(f"{Fore.GREEN}[50%]{Fore.WHITE}Importing..")  
   
 #Cleaning  
 df = dataset[dataset["Current Loan Amount"] < 999999 ]  
 clean\_dataset = df.drop\_duplicates(subset='Customer ID')  
  
 if x == 75:  
 print(f"{Fore.GREEN}[75%]{Fore.WHITE}Perform Calculations..")  
  
 #get calculation  
 scored = cal.getscore(dataset)  
 #get classification  
 final = classi.byRandomForest(scored)  
  
  
 if x == 99:  
 ###-----ADDING NEW COLUMNS WITH CALCULATED VALUES TO CSV-----###  
 ###------------SAVING CALCULATED CSV WITH SCORES-------------###  
   
 #to display all output values  
 pd.set\_option("display.max\_rows", None, "display.max\_columns", None)  
  
 # replace NaN values with 0  
 final= final.fillna(0)  
  
 final.to\_csv('data/Dataset\_Scored.csv', index=False)  
  
 print(f"{Fore.GREEN}[100%]{Fore.WHITE}Dataset Successfully loaded")  
 bar()  
  
print(Fore.GREEN +"\n\n|---------------------SUCCESSFULLY LOADED--------------------|")  
#######-----------END OF LOADING PROGRESS BAR--------########

Importing Dataset :  
  
on 25: [25%]Checking Dataset..  
on 50: [50%]Importing..  
on 75: [75%]Perform Calculations..  
on 99: [100%]Dataset Successfully loaded  
|████████████████████████████████████████| 100/100 [100%] in 15.1s (6.64/s)   
  
  
|---------------------SUCCESSFULLY LOADED--------------------|

ASKING USER INPUT FOR THE ALGORITHM FEATURES

######------ Options list to command------######  
tool.options\_list()  
Exit = False  
#2. Ask for option and validate if valid input or not.  
while Exit == False:  
 try:  
 input1 = input("\nSelect Option:")   
  
 if input1.capitalize() == "A":  
 ###---Loan Info---###  
 print(f"\n{Fore.GREEN}You've Choosen option A")  
 print(f"\n{Fore.GREEN}CUSTOMER ID LIST:")  
  
 #1. Display CUSTOMER ID  
 with open('data/Dataset\_Scored.csv', newline='') as csvfile:  
 newdataset1 = csv.DictReader(csvfile)  
 print("Customer ID")  
 print("--------------------")  
 for row in newdataset1:  
 print(row['Customer ID'])  
 tool.askidshow()  
  
 #4. Display options after the while loop ended  
 sleep(2.60)  
 tool.options\_list()  
 continue  
 elif input1.capitalize() == "B":  
 ###---Show CScore---###  
 print(f"\n{Fore.GREEN}You've Choosen option B\n\n")  
 print(f"\n{Fore.GREEN}CREDIT SCORES LIST:")  
 #1.display  
 with open('data/Dataset\_Scored.csv', newline='') as csvfile:  
 newdataset2 = csv.DictReader(csvfile)  
 print("Customer ID | Credit Score | Class Score")  
 print("---------------------------------")  
 for row in newdataset2:  
 print(row['Customer ID']," ", row['Score']," ",row['Class Score'])  
   
 elif input1.capitalize() == "C":   
 ###---CScore BAR Graph---###  
 print(f"\n{Fore.GREEN}You've Choosen option C")  
 print(f"\n{Fore.GREEN}BAR GRAPH:")  
 Number\_of\_Customer = classi.obtain(pd.read\_csv('data/Dataset\_Scored.csv'))  
 tool.bar\_chart(Number\_of\_Customer)  
 print(f"\n{Fore.RED}Graph Exited. Returning...\n\n")  
 sleep(2.00)  
  
 tool.options\_list()  
 continue  
  
 elif input1.capitalize() == "E":   
 ###---About CSPA---###  
 print(f"\n{Fore.GREEN}You've Choosen option E")  
 print(f"\n{Fore.WHITE}CSPA by CSWarriors V.1")  
 osCommandString = "notepad.exe About\_CSPA.txt"  
 os.system(osCommandString)  
  
 elif input1.capitalize() == "D":  
 ###---New Data---###  
 print(f"\n{Fore.GREEN}You've Choosen option D")  
  
 field\_names = [['Customer ID', 'Current Loan Amount', 'Current Credit Balance',  
 'Monthly Debt', 'Years in current job','Years of Credit History',  
 'Number of Open Accounts', 'Months since last delinquent', 'Maximum Open Credit'],]  
  
 with open('data/Dataset\_Scored.csv','a', newline='') as csvfile:   
  
 # Gather new user data  
 dict = tool.newuser(dataset.iloc[-1,0])  
   
 # Pass the dictionary as an argument to the Writerow()  
 dictwriter\_object = DictWriter(csvfile, fieldnames=field\_names[0])  
 dictwriter\_object.writerow(dict)  
  
 dataset = pd.read\_csv('data/Dataset\_Scored.csv')  
 #get calculations  
 scored = cal.getscore(dataset)  
  
 #get classification  
 final = classi.byCSPA(scored)  
 final.to\_csv('data/Dataset\_Scored.csv', index=False)  
   
 elif input1.capitalize() == "F":  
  
 print(f"\n{Fore.GREEN}You've Choosen option F")  
 exitinputE = input("\nAre you sure you want to exit? (Y/N):")   
 if exitinputE.capitalize() == "N":  
 print(f"\n{Fore.GREEN}Returning...\n\n")  
 tool.options\_list()  
 continue  
 elif exitinputE.capitalize() == "Y":  
 print(f"\n{Fore.RED}Closing...\n\n")  
 Exit == True  
 break  
 else:  
 print(f"\n{Fore.RED}Invalid input, returning... \n\n")  
 #H.1 IF outside the options, then ask again  
 else:  
 input1 != "A" or "B" or "C" or "D" or "E" or "F"  
 print(f"\n{Fore.RED}Input is invalid.{Fore.GREEN} Please choose only on the options provided.")  
 continue  
 except:  
 continue

PLEASE CHOOSE AN OPTION!  
  
  
[A] View Customer Info.  
  
[B] Show Credit Score.  
  
[C] Check Credit Score Bar Graph.  
  
[D] Enter new user data.  
  
[E] About the CSPA.  
  
[F] Exit.  
  
You've Choosen option F  
  
Closing...