Assignment 1: Comp 2067

Submission Date 2023-06-27

Student ID: 110035023
Student Name: Michael Joyce

II. Exercises

Exercise 1: Number Pattern Write a function named number_pattern that takes a positive integer n as input and prints the following pattern: The number of rows in the pattern should be equal to n. Use loops and conditional statements to generate the pattern. Test your function with different values of n.

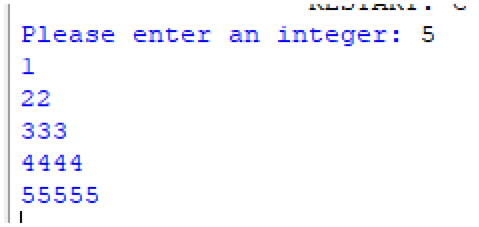
Exercise 2: Rock-Paper-Scissors Game Write a function named rock_paper_scissors that allows a user to play the classic game of RockPaper-Scissors against the computer. The function should prompt the user for their choice (rock, paper, or scissors), generate a random choice for the computer, compare the choices, and determine the winner. Use loops, conditional statements, and functions to implement the game. Test your function by playing the game multiple times.

Exercise 3: Banking Application Write a program that simulates a simple banking application. The program should allow users to perform basic banking operations such as account balance inquiry, depositing funds, and withdrawing funds. The program should also keep track of multiple user accounts. Requirements: 1. Implement a function to create a new bank account with an initial balance. 2. Implement a function to display the account balance for a given account number. 3. Implement a function to deposit funds into a specific account. 4. Implement a function to withdraw funds from a specific account, ensuring that the account has sufficient balance. 5. Use a loop to continuously prompt the user for operations until they choose to exit.

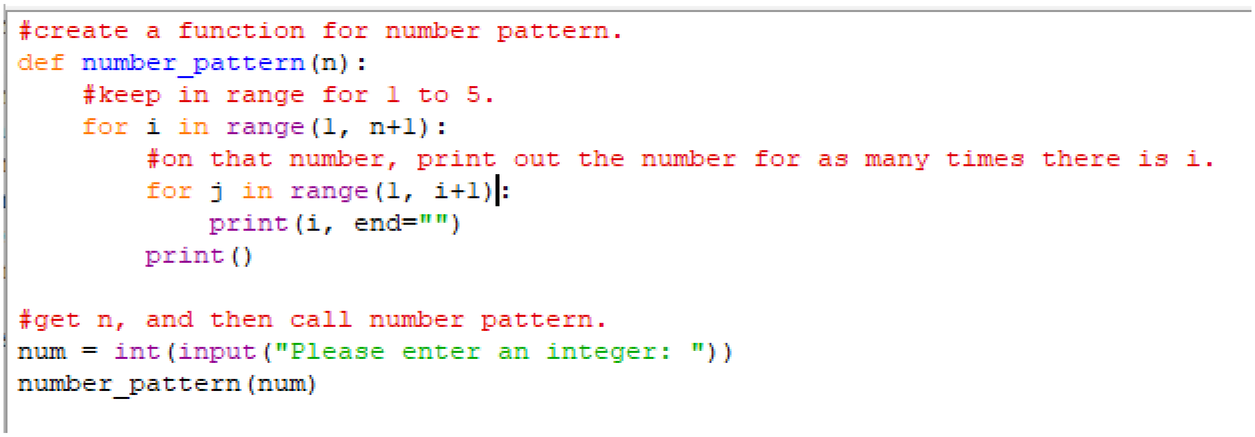
Codes and Results

Exercise 1

Enter a number, it will output that number, for as many digits as the row number.

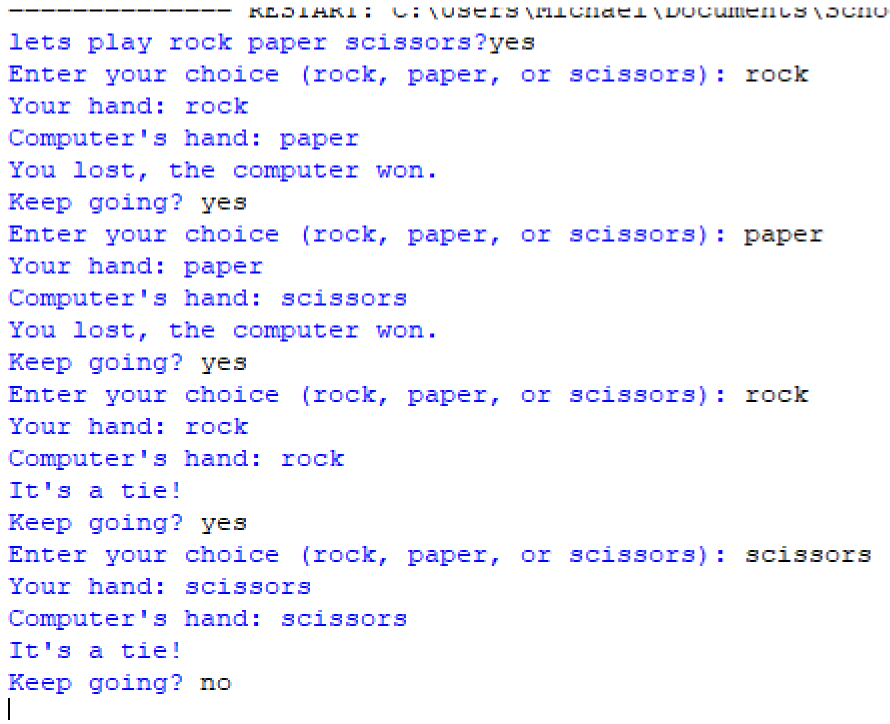


```
Please enter an integer: 5
1
22
333
4444
55555
```

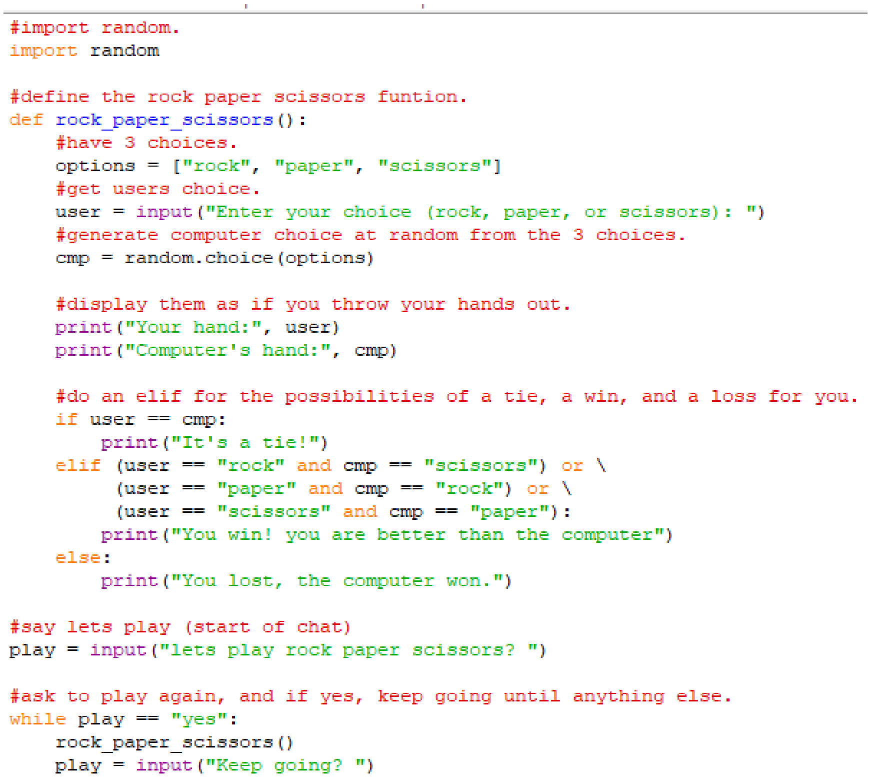


```
#create a function for number pattern.
def number_pattern(n):
    #keep in range for 1 to 5.
    for i in range(1, n+1):
        #on that number, print out the number for as many times there is i.
        for j in range(1, i+1):
            print(i, end="")
        print()

#get n, and then call number pattern.
num = int(input("Please enter an integer: "))
number_pattern(num)
```

Excerise 2
Enter a choice of rock paper or scissors, and the computer will do the same at random. Depending on the choice, you will win lose or tie. Continue playing by typing in yes.



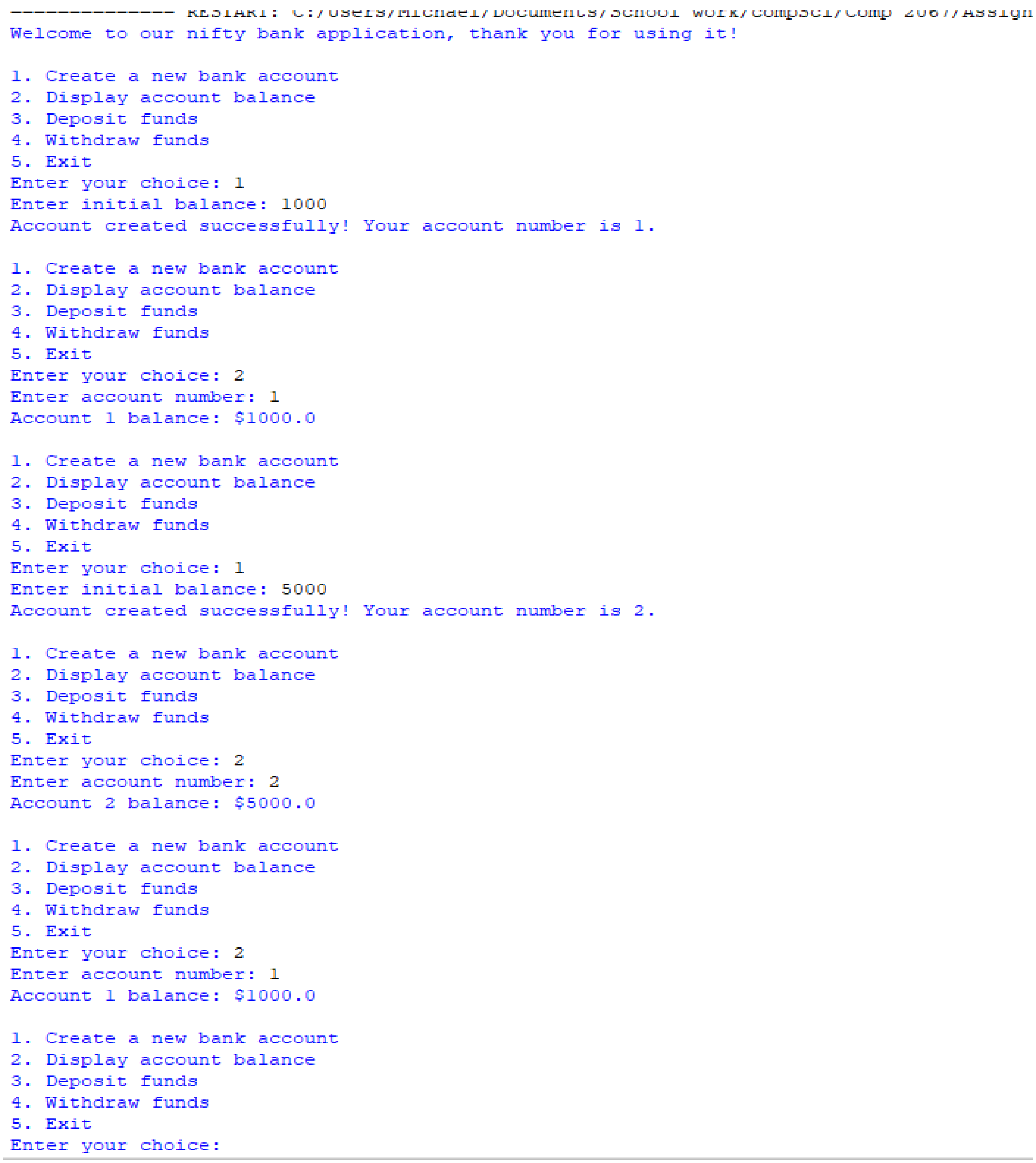
```
================ RESTART: C:\Users\Michael\Documents\School
lets play rock paper scissors?yes
Enter your choice (rock, paper, or scissors): rock
Your hand: rock
Computer's hand: paper
You lost, the computer won.
Keep going? yes
Enter your choice (rock, paper, or scissors): paper
Your hand: paper
Computer's hand: scissors
You lost, the computer won.
Keep going? yes
Enter your choice (rock, paper, or scissors): rock
Your hand: rock
Computer's hand: rock
It's a tie!
Keep going? yes
Enter your choice (rock, paper, or scissors): scissors
Your hand: scissors
Computer's hand: scissors
It's a tie!
Keep going? no
|
```



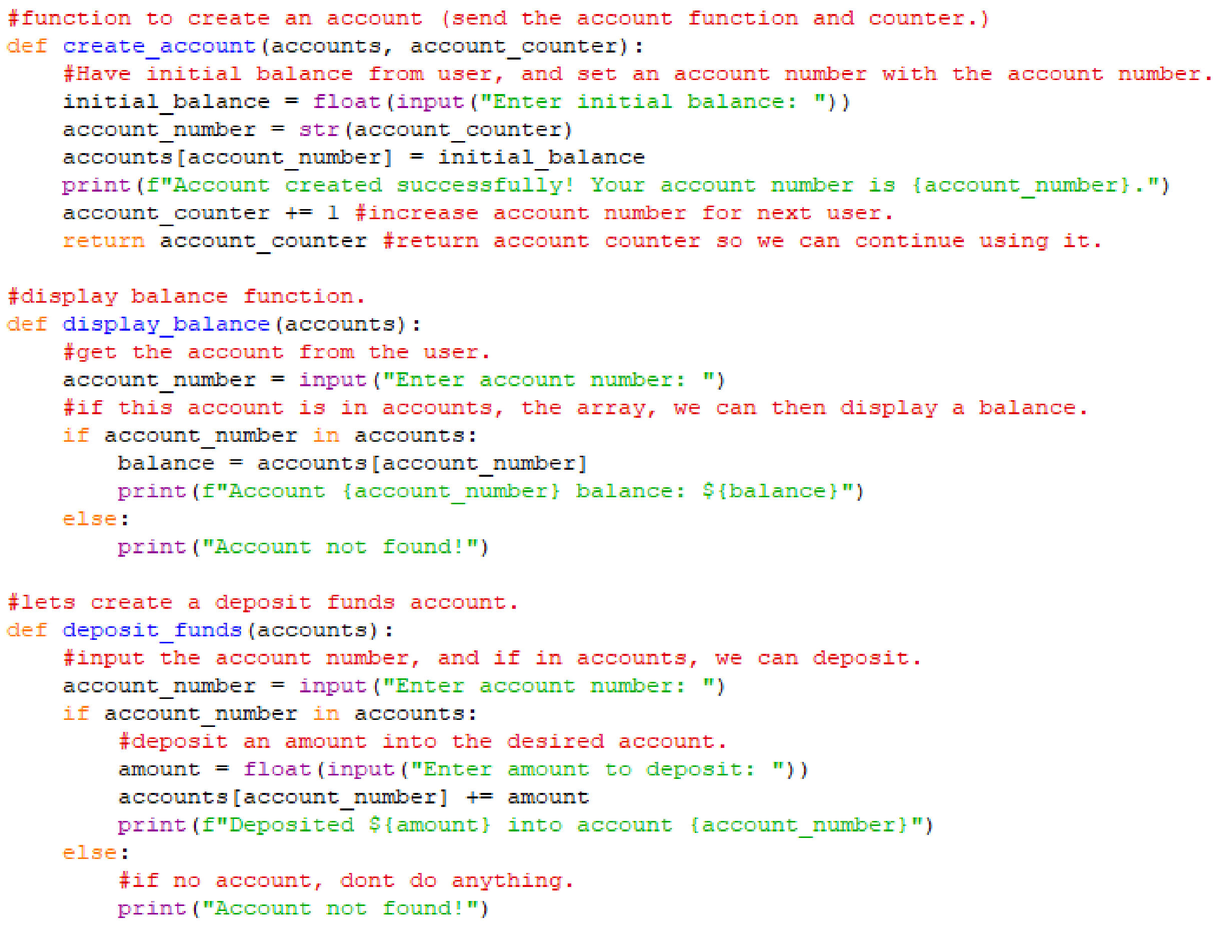
```
#import random.
import random

#define the rock paper scissors funtion.
def rock_paper_scissors():
    #have 3 choices.
    options = ["rock", "paper", "scissors"]
    #get users choice.
    user = input("Enter your choice (rock, paper, or scissors): ")
    #generate computer choice at random from the 3 choices.
    cmp = random.choice(options)

    #display them as if you throw your hands out.
    print("Your hand:", user)
    print("Computer's hand:", cmp)

    #do an elif for the possibilities of a tie, a win, and a loss for you.
    if user == cmp:
        print("It's a tie!")
    elif (user == "rock" and cmp == "scissors") or \
         (user == "paper" and cmp == "rock") or \
         (user == "scissors" and cmp == "paper"):
        print("You win! you are better than the computer")
    else:
        print("You lost, the computer won.")

#say lets play (start of chat)
play = input("lets play rock paper scissors? ")

#ask to play again, and if yes, keep going until anything else.
while play == "yes":
    rock_paper_scissors()
    play = input("Keep going? ")
```

Excerise 3:

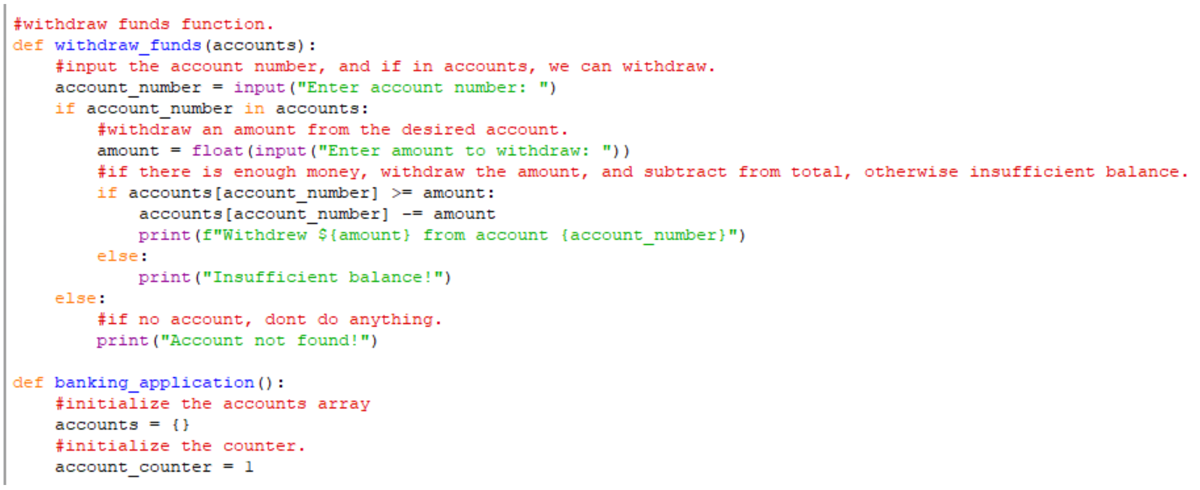
Have a bank application. Create an account by choosing option 1. It is tracked in an array, and add a balance to it. Can then check balance, add to it, and then withdraw from it using your account number that is provided. A new account number will be increased by 1 every time someone creates one.



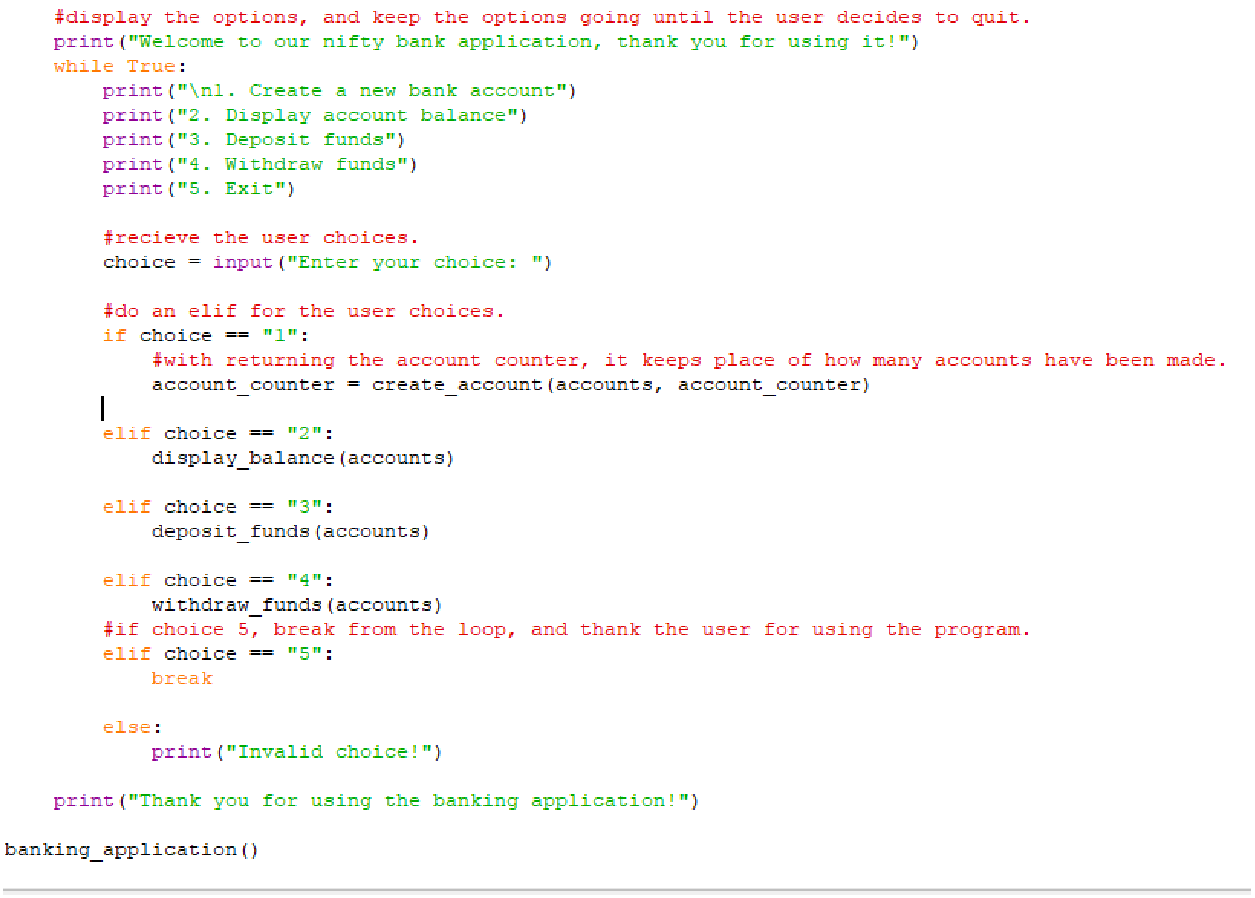
```
============== RESTART: C:/Users/Michael/Documents/School Work/CompSci/Comp 2067/Assign
Welcome to our nifty bank application, thank you for using it!

1. Create a new bank account
2. Display account balance
3. Deposit funds
4. Withdraw funds
5. Exit
Enter your choice: 1
Enter initial balance: 1000
Account created successfully! Your account number is 1.

1. Create a new bank account
2. Display account balance
3. Deposit funds
4. Withdraw funds
5. Exit
Enter your choice: 2
Enter account number: 1
Account 1 balance: $1000.0

1. Create a new bank account
2. Display account balance
3. Deposit funds
4. Withdraw funds
5. Exit
Enter your choice: 1
Enter initial balance: 5000
Account created successfully! Your account number is 2.

1. Create a new bank account
2. Display account balance
3. Deposit funds
4. Withdraw funds
5. Exit
Enter your choice: 2
Enter account number: 2
Account 2 balance: $5000.0

1. Create a new bank account
2. Display account balance
3. Deposit funds
4. Withdraw funds
5. Exit
Enter your choice: 2
Enter account number: 1
Account 1 balance: $1000.0

1. Create a new bank account
2. Display account balance
3. Deposit funds
4. Withdraw funds
5. Exit
Enter your choice:
```



```
#function to create an account (send the account function and counter.)
def create_account(accounts, account_counter):
    #Have initial balance from user, and set an account number with the account number.
    initial_balance = float(input("Enter initial balance: "))
    account_number = str(account_counter)
    accounts[account_number] = initial_balance
    print(f"Account created successfully! Your account number is {account_number}.")
    account_counter += 1 #increase account number for next user.
    return account_counter #return account counter so we can continue using it.

#display balance function.
def display_balance(accounts):
    #get the account from the user.
    account_number = input("Enter account number: ")
    #if this account is in accounts, the array, we can then display a balance.
    if account_number in accounts:
        balance = accounts[account_number]
        print(f"Account {account_number} balance: ${balance}")
    else:
        print("Account not found!")

#lets create a deposit funds account.
def deposit_funds(accounts):
    #input the account number, and if in accounts, we can deposit.
    account_number = input("Enter account number: ")
    if account_number in accounts:
        #deposit an amount into the desired account.
        amount = float(input("Enter amount to deposit: "))
        accounts[account_number] += amount
        print(f"Deposited ${amount} into account {account_number}")
    else:
        #if no account, dont do anything.
        print("Account not found!")
```



```
#withdraw funds function.
def withdraw_funds(accounts):
    #input the account number, and if in accounts, we can withdraw.
    account_number = input("Enter account number: ")
    if account_number in accounts:
        #withdraw an amount from the desired account.
        amount = float(input("Enter amount to withdraw: "))
        #if there is enough money, withdraw the amount, and subtract from total, otherwise insufficient balance.
        if accounts[account_number] >= amount:
            accounts[account_number] -= amount
            print(f"Withdrew ${amount} from account {account_number}")
        else:
            print("Insufficient balance!")
    else:
        #if no account, dont do anything.
        print("Account not found!")

def banking_application():
    #initialize the accounts array
    accounts = {}
    #initialize the counter.
    account_counter = 1
```



```
    #display the options, and keep the options going until the user decides to quit.
    print("Welcome to our nifty bank application, thank you for using it!")
    while True:
        print("\n1. Create a new bank account")
        print("2. Display account balance")
        print("3. Deposit funds")
        print("4. Withdraw funds")
        print("5. Exit")

        #recieve the user choices.
        choice = input("Enter your choice: ")

        #do an elif for the user choices.
        if choice == "1":
            #with returning the account counter, it keeps place of how many accounts have been made.
            account_counter = create_account(accounts, account_counter)

        elif choice == "2":
            display_balance(accounts)

        elif choice == "3":
            deposit_funds(accounts)

        elif choice == "4":
            withdraw_funds(accounts)
        #if choice 5, break from the loop, and thank the user for using the program.
        elif choice == "5":
            break

        else:
            print("Invalid choice!")

    print("Thank you for using the banking application!")

banking_application()
```