- **Object-Oriented Programming Assignment 2**

Description of BankAccount.py
Defined class name BankAccount with 5 attributes : account_number, account_type, balance, account_name(name) and password
If account_type = 10 (Saving account)
Elif account_type = 20 (Checking account)
Elif account_type = 30 (Brokerage account)
Else account_type = Saving account
Input balance have to more than 0
And all attributes are encapsulate

```
class BankAccount:
    def __init__(self, account_number, account_type, balance, name, password):
        self.__account_number = account_number
        if account_type == 10:
            self.__account_type = "Saving account"
        elif account_type == 20:
            self.__account_type = "Checking account"
        elif account_type == 30:
            self.__account_type = "Brokerage account"
        else:
            self.__account_type = "Saving account"
        if balance >= 0:
            self.__balance = balance
        else:
            print("Invalid amount!")
        self.__name = name
        self.__password = password
```

In this class have 10 methods (not include __init__)

1.) Deposit methods have 2 argument amount and password

If you want to deposit money, the password has to be corrected, If not the message will show “Password is not correct”, Then check the amount value has to be more than 0 (if less than 0 the message will show “You can’t deposit a negative of amount!”) so you can deposit money to BankAccount and show the message “Your current balance is ….(depending on the balance of your BankAccount)”.

```
    def deposit(self, amount, password):
        if self.__password != password:
            print("Password is not correct!\n")
        else:
            if amount >= 0:
                self.__balance = self.__balance + amount
                print("Your current balance is " + str(self.get_balance()) + "\n")
            else:
                print("You can't deposit a negative of amount!\n")
```

2.) Withdraw methods have 2 argument amount and password

If you want to withdraw money, the password has to be corrected, If not the message will show “Password is not correct”, Then check the amount value has to be more than 0 and less than self.__balance

```
def withdraw(self, amount, password):
    if self.__password != password:
        print("Password is not correct!\n")
    else:
        if (amount > self.__balance) and (amount >= 0):
            print("Your current balance is not enough, Please try again.\n")
        elif amount < 0:
            print("You can't withdraw a negative of amount!\n")
        else:
            self.__balance = self.__balance - amount
            print("Your current balance is " + str(self.get_balance()) + "\n")
```

3.) Transfer methods have 3 argument account, amount and password

If you want to transfer money, the password has to be corrected, If not the message will show “Password is not correct” then deposit money to destination account and withdraw money from self

```
def transfer(self, account, amount, password):
    if self.__password != password:
        print("Password is not correct!\n")
    else:
        if (self.__balance >= amount) and (amount >= 0) :
            self.__balance = self.__balance - amount
            account.__balance = account.__balance + amount
            print(f"You transfered {amount} to {account.__account_number}. \n")
        elif amount > self.__balance:
            print("You don't have enough money!\n")
        elif amount < 0:
            print("You can't transfer a negative of amount! \n")
```

4.) callInterest methods check the account_type of self

```
def calInterest(self):
    if self.__account_type == "Saving account":
        interestRate = 1.08
    elif self.__account_type == "Checking account":
        interestRate = 1.02
    elif self.__account_type == "Brokerage account":
        interestRate = 1.15
    else:
        interestRate = 1.00
    print("Your account interest rate is " + str(interestRate) + "\n")
```

5.) _acc_info methods has 1 argument is password

If you want to see the information of account, the password has to be corrected, If not the message will show “Password is not correct” , Then show all the information of self.account

```
def _acc_info(self, password):
    if self.__password != password:
        print("Password is not correct! \n")
    else:
        print(f"Account number : " + str(self.get_account_number()) + ", Account type : " + str(self.get_account_type()) +
            ", Account name : " + str(self.get_name()) + ", Balance : " + str(self.get_balance()) +"\n")
```

6.) Get_account_number return self.__account_number

```
def get_account_number(self):
    return self.__account_number
```

7.) Get_account_type return self.__account_type

```
def get_account_type(self):
    return self.__account_type
```

8.) Get_balance return self.__balance

```
def get_balance(self):
    return self.__balance
```

9.) Get_name return self.__name

```
def get_name(self):
    return self.__name
```

10.) Get_password return self._password

```
def get_password(self):
    return self.__password
```

- **Example in main.py**
  Woraphob Sinbunyama 630910359