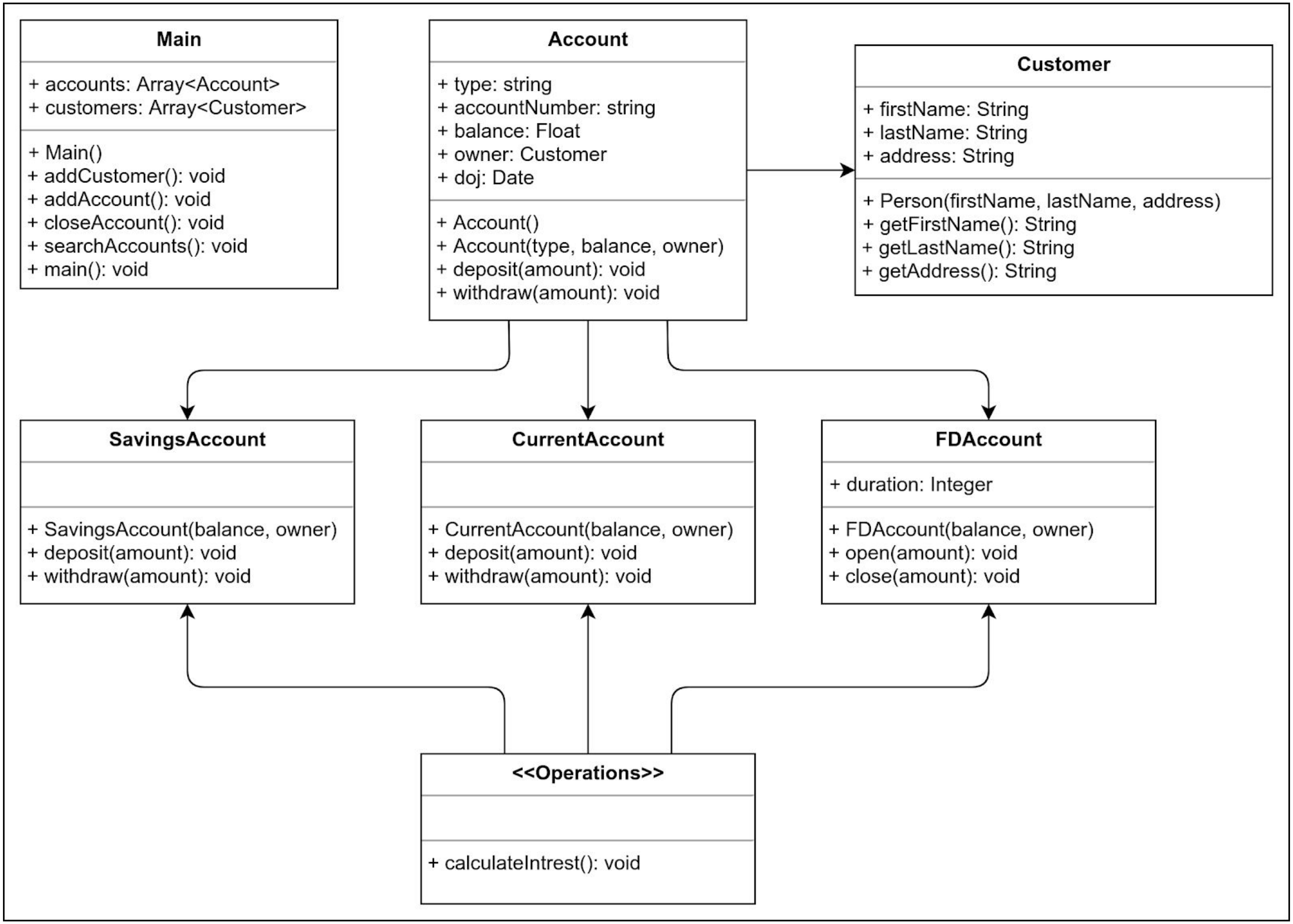
Class Diagram



Main

+ accounts: Array<Account>
+ customers: Array<Customer>

+ Main()
+ addCustomer(): void
+ addAccount(): void
+ closeAccount(): void
+ searchAccounts(): void
+ main(): void

Account

+ type: string
+ accountNumber: string
+ balance: Float
+ owner: Customer
+ doj: Date

+ Account()
+ Account(type, balance, owner)
+ deposit(amount): void
+ withdraw(amount): void

Customer

+ firstName: String
+ lastName: String
+ address: String

+ Person(firstName, lastName, address)
+ getFirstName(): String
+ getLastName(): String
+ getAddress(): String

SavingsAccount

+ SavingsAccount(balance, owner)
+ deposit(amount): void
+ withdraw(amount): void

CurrentAccount

+ CurrentAccount(balance, owner)
+ deposit(amount): void
+ withdraw(amount): void

FDAccount

+ duration: Integer

+ FDAccount(balance, owner)
+ open(amount): void
+ close(amount): void

<<Operations>>

+ calculateIntrest(): void

The details for each account are provided below -

1. Savings account is providing interests 4%
2. Current account is providing interests 0%
3. FD account is providing interests 6.45%

Complete the code in the editor below to implement the following:

1. Account can be owned by customer
2. Customer and account needs to be maintained in separate array
3. Create a Sample Program following conditions
    a. customers Array should be able to store 30 Customers
    b. accounts Array should be able to store 60 Accounts
    c. Ask user for input, 1 for add customer, 2 add account, 3 for close account, 4 search account, 5 calculate interest, 6 deposit, 7 widrawal. (Refer Hints#1)
        i. For 1, Ask next level of input for customer details -> create the object and store in the customers array
        ii. For 2, Ask for account details, auto generate account number,store it in accounts details, print account details in console
        Note: search customer by firstName and lastName and link with customer reference created in step 1
        iii. For 3, Ask for accountNumber and delete account from array
        iv. For 4 -> ask for customer's firstName and lastName -> print all account details related to customer
        Example
        Accounts of Riddhi Gajera
        1. AC00015 Savings, 10000 rs, 10-Jan-2020
        v. For 5 -> ask for customer's firstName and lastName and calculate interest of all accounts owned by customer and update balance of account
        vi. For 6, ask for account number and amount and deposit the amount
        vii. For 7, ask for account number and amount, check the balance is sufficient and withdraw the amount

Constraints:

1. You can use java, python or javascript language for exercise
2. Account should not be initiated by new Account()
3. Each class must be defined in separate file
4. Main file name must be Main.java/Main.py/Main.js
5. You need to push the code under gitlab
6. accountNumber in Account model needs to unique
7. firstName and lastName combination in Customer model needs to unique

8. Single customer can own multiple accounts
9. You can use Sublime, Atom and Visual Studio code without any extensions for this exercise.
10. Create README.MD for providing the instructions how to execute the program

Hints:

1. Keep Main class as short as possible
2. You can add custom methods in any model if you feel need so
3. Taking User inputs

java	https://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html
	example
	Scanner input = new Scanner(System.in);
	int number = input.nextInt();
	https://docs.oracle.com/javase/7/docs/api/java/io/InputStreamReader.html
	BufferedReader br = new BufferedReader(new
	InputStreamReader(System.in))
	String name = br.readLine();

Javascript	For Browser: you can use prompt
	Example: var username = prompt("What is your name?");
	For NodeJs:
	// Get process.stdin as the standard input object.
	var standard_input = process.stdin;
	// Set input character encoding.
	standard_input.setEncoding('utf-8');
	// Prompt user to input data in console.
	console.log("Please input text in command line.");
	// When user input data and click the enter key.

	standard_input.on('data', function (data) {
	
	// User input exit.
	if(data === 'exit\n'){
	// Program exit.
	console.log("User input complete, program exit.");
	process.exit();
	}else
	{
	// Print user input in console.
	console.log('User Input Data : ' + data);
	}
	});

Python	g = raw_input("Enter your name : ")
	print g
	
	num = input ("Enter number :")
	print(num)
	name1 = input("Enter name : ")
	print(name1)

Answer the following questions:

1. How many classes have you created?
2. Where have you used Object?
3. Where have you used the same object reference in multiple places?
4. What part explains runtime Polymorphism?
5. What part explains Data Hiding Concepts? Why?
6. Where you have used Abstraction and what is the use of that?
7. What kind of Inheritance you have implemented

8. What part explains static/compile-time Polymorphism?
9. What explains the method overloading in your example?
10. While calling withdraw which method will be called in case of SavingsAccount?
11. What part explains the constructor overloading?