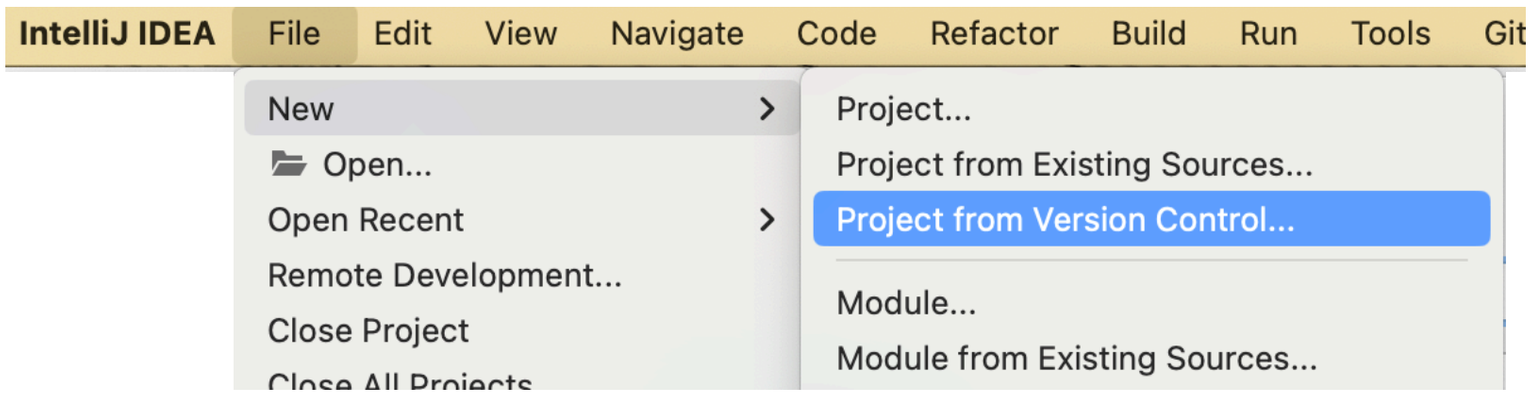
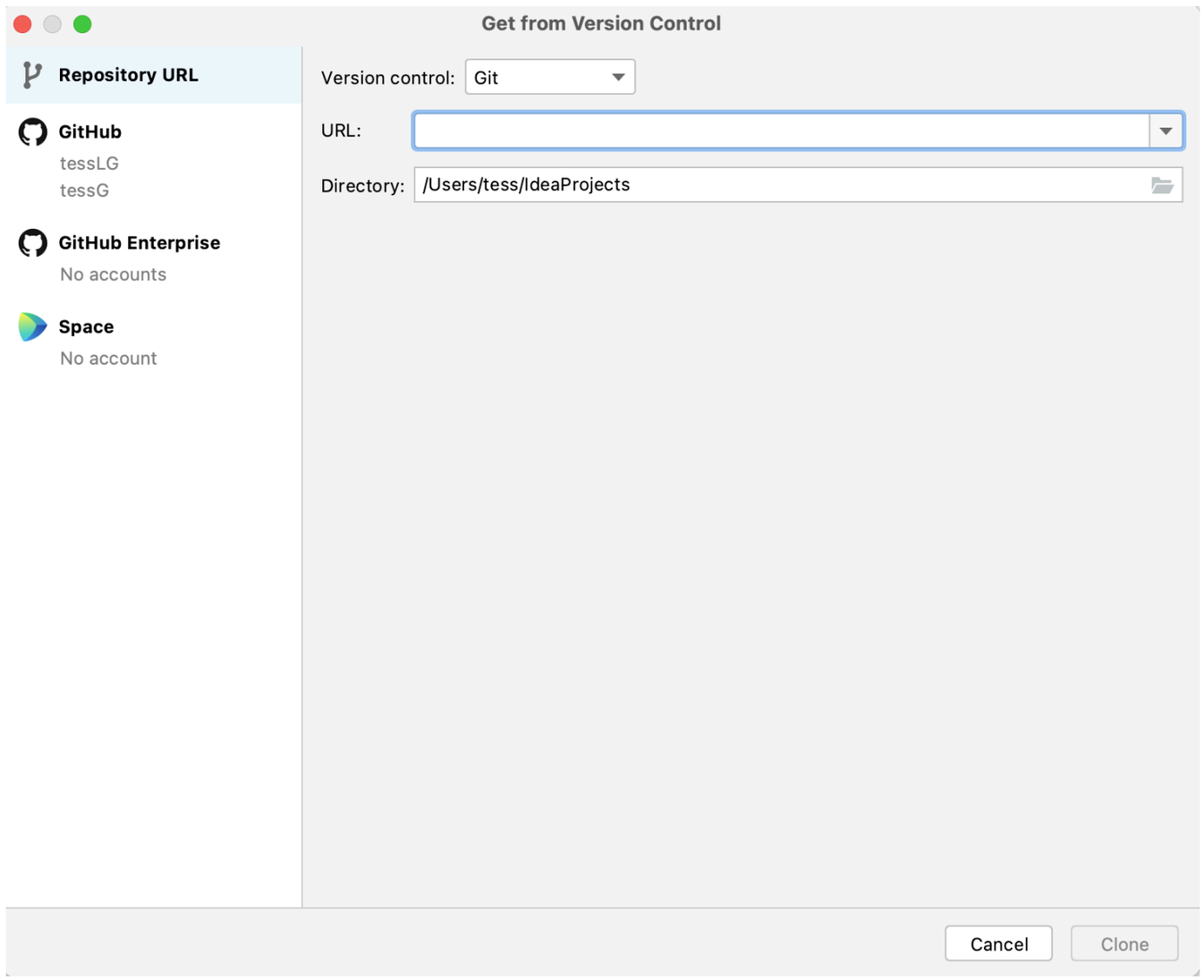
# Start et nyt projekt fra git repo




https://github.com/Dat1Cphbusiness/BankSystem.git




**Obs!**

Kald projektet **TurnbasedGame**

# SPRINT1 turnbasedGame

## Developer A

Rename `Customer` class to `Player`.

Remember, construtor, class and File must have the exact same name

(Tip: use the refactor command, select all suggested changes in the pop-up)

Rename ‘customer.csv’ to ‘playerData.csv’
The value of path attribute in the Main class must reflect this change.

---

1.In the `Game` class, add method `registerPlayers` to the `Game` class.
2. Clear the `players` list by instantiating again.
3. Inside it, write a `while` loop that runs as long as the size of the `players` list is less than 6
4. Inside the `while` body, call the `createPlayer` method, with arguments `null` and `0`

---

In the `switch-case` of the `runDialog`, we now have 3 cases. For now, we only need case 1 and case 2.

Start by deleting case 3, and the content of case 1 and 2. (Keep the break command).

In case 1 (start new game), call `registerPlayers`.
In case 2 (continue game ), call `loadPlayerData`, which you will implement later.
(You may out-comment the call, or add the method with an empty body)

Add the `runGameLoop` method to the `Game` class. This should be called from each of the 2 cases. Implement it in the next step…

## Developer B

In the `Bank` class,
1. rename class `Bank` -> `Game`
2. If necessary, change `ArrayList<Customer> customers` -> `ArrayList<Player> players`
3. change menu items to

```
"1) start new game”
"2) continue game"
"3) quit game”
```

4. rename `createCustomer -> createPlayer`, adjust wording of the message
5. rename `displayCustomers -> displayPlayers`. If necessary, change the body of the code accordingly.

---

In the `Game` class add the remaining attributes as per the class diagram:

`TextUI ui` - cut and paste from Main
`FileIO io` - out comment this, as you have yet to create `FileIO`
`String playerDataPath` - cut and paste from Main

---

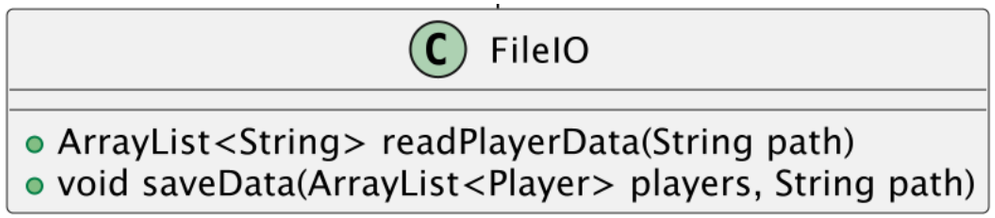
1. create local variable `count` of type `int` with the value `0`
2. create a local variable `input` of type String with the value `"Y"`
3. add a `while`-loop that runs as long as `input` is `“Y"`
*Inside the loop:*
4. set `currentPlayer` to the `player` at index `count` in the `players` list
5. display a message to inform who's turn it is (use the `ui` instance)
6. make a call to a method we have not yet made called `throwAndMove`. Out-comment this call for now…
7. prompt the user with the message `“Fortsæt? Y/N”` using `ui.promptText`, assign the return value to `input`
8. increment the `count` variable by 1.
9 use `count` to check if we have reached the end of the `players` list. if so, reset it.

# SPRINT2
# turnbasedGame w. FileIO

## Developer A

Create the class `FileIO`, such that it reflects the class diagram.

- leave the methods empty for now…




Cut those lines of code in `main`, where data is read. This should leave you with two lines of code.
Paste the lines into the `readPlayerData` method in `FileIO`.

Cut the code from inside the `saveData` of the Main class and paste it into the `saveData` in the `FileIO` class.

Change the signature such that the method receives both the list of players and the path to the `playerData.csv` file

Add an `if-else` statement with a condition that checks if data has any elements.

In the `else` part of the statement:
Call the `registerPlayers` method

## Developer B

We need to change the code in `readPlayerData` a bit, to make it more generic.

In the first line of the method, initialize an `ArrayList<String> data`. (We will eventually return this list.)
In the while loop, after the first line that scans the next line, add 1 line of code that adds the `line` to our list, `data`.

Delete or out-comment the last 4 lines of code, (we will do what they do some where else).

Return the `data` list.

In the `Game` class, we will now implement the `loadPlayerData` method, which is being called from `case 2` of the `runDialog` method:

In-comment the `FileIO` instantiation made in previous sprint.

In the body of `loadPlayerData` create a local variable `ArrayList<String> data` and assign to it, the value returned from a call to `readPlayerData` on the `io` object:

```
ArrayList<String> data = io.readPlayerData(playerDataPath);
```

In the `if` part of the statement:
Create a loop: For each element `s` in `data`, use the `split` method to separate the values: `String[] values = s.split(",");`
Then call the `createPlayer` method using the elements in the `values` array as arguments.

You may reuse the lines you out-commented in step2.