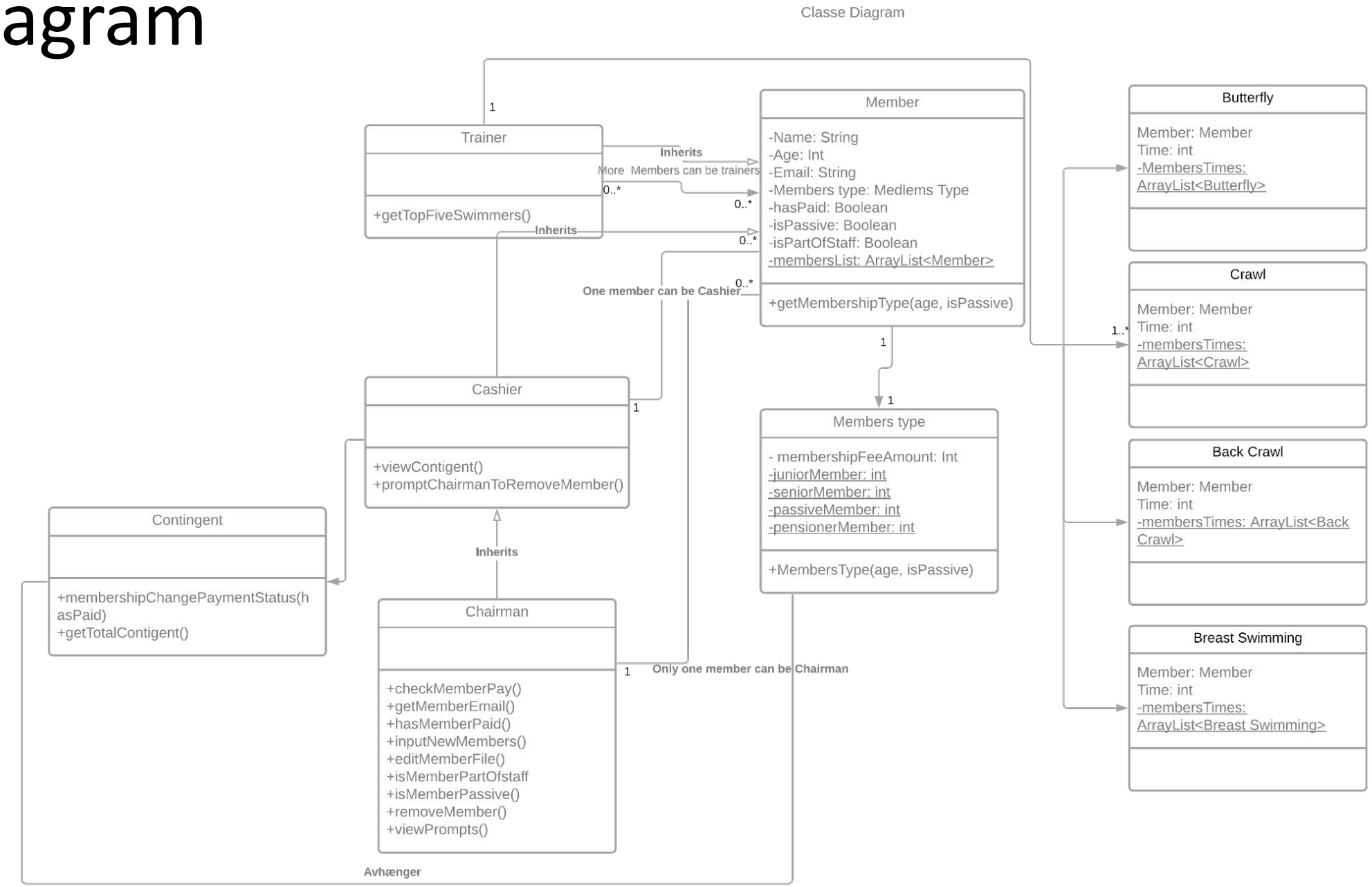
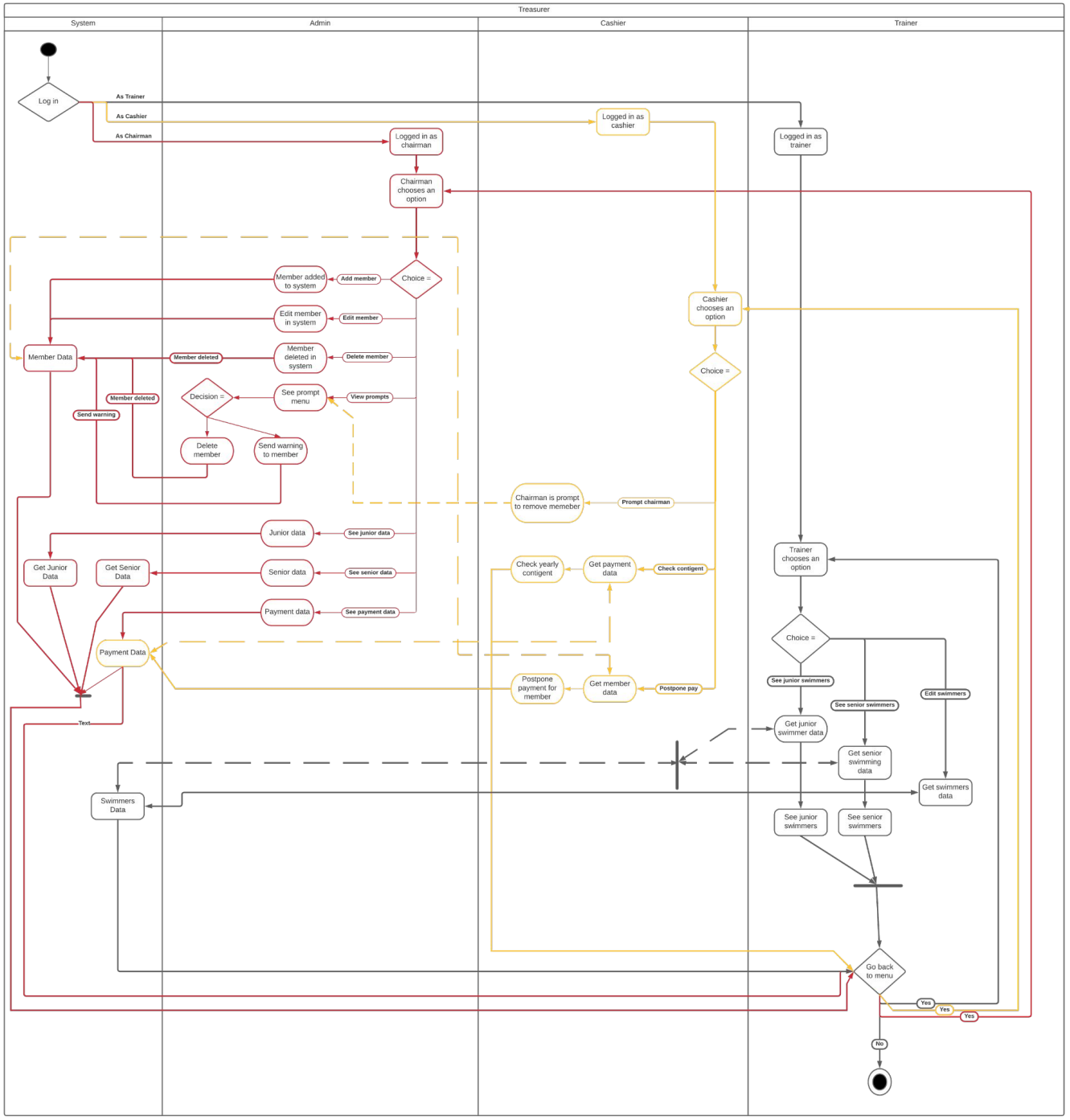
Delfinen

Gruppe 3: De seje hunde ʕ•ᴥ•ʔ

Class diagram



Activity diagram



Logical breakdown of related classes into packages

- **We have decided to make five packages**
  - FileWorkers
  - MembersClasses
  - TestThings
  - TrainingGroup
  - UI

- **Example UI**
  - E. g. the CashierUI is the user interface for the cashier
  - Here, the cashier can see everything he/she can interact with and his/her possiblities

GRASP principles in our code

The creator

```java
public class MembersType
{
    // Setting up price variables for different member types
    private static int juniorMember = 1000;
    private static int seniorMember = 1600;

    // Calculating the pensioner member price based on the senior member price (i
    private static int pensionerMember = (int) Math.round(seniorMember * .75);
    private static int passiveMember = 500;
    private int yearlySubscriptionPrice;
```

```java
public MembersType(int age, boolean isPassive)
{
    if (age < 18 && !isPassive)
    {
        yearlySubscriptionPrice = juniorMember;
        memberType = "juniorMember";
    }

    // Else if member is +18 and under 60 not passive
    else if (age >= 18 && age < 60 && !isPassive)
    {
        yearlySubscriptionPrice = seniorMember;
        memberType = "seniorMember";
    }
```

The Information Expert

```java
// Creating Member class
public class Member
{
    // Creating a list for the members
    private static ArrayList<Member> membersList = new ArrayList<>();

    // Instantiating name
    private String name;

    // Instantiating age
    private int age;

    // Instantiating e-mail
    private String email;

    // Instantiating MembersType
    private MembersType membersType;

    // Instantiating hasPaid boolean
    private boolean hasPaid;
```

```java
public Member(String name, int age, String email, boolean hasPaid, boolean isPassive, boolean isPartOfStaff)
{
    this.name = name;
    this.age = age;
    this.email = email;
    this.hasPaid = hasPaid;
    this.isPartOfStaff = false;
    this.isPassive = false;
    this.membersType = getMemberShipType();
}
```

Use of Low coupling

We are able to create new classes in our code as long as:

connect them to system

```
switch (loginUser)
{
    // 1. Break
    case 1:
        ChooseDisciplineUI.chooseSwimmingDiscipline();
        break;
    // 2. Calling the cashierMenu method
    case 2:
        CashierUI.cashierMenu();
        break;

    // 3. Calling the chairmanMenu method
    case 3:
        Chairman.chairmanMenu();
        break;
}
```

Relevant data structures (Array and ArrayList)

- **Arrays are used when number of stored data elements is known**
- **ArrayLists are used when number of stored data elements are unknown**

- **ArrayList example (used for storing the members as they're being created):**

```
private ArrayList<Member> members = new ArrayList();
```

- **Array example (used for the name generator and used for testing a function):**

```
String[] names = {"Nicolas", "Lasse", "Tobias", "Harald", "Carl", "Gudit", "Erik"};
```

Ability to store relevant data in file(s)

- The FileWriter and the FileReader class are imported to use to read and write from files
- Data is stored in and read from CSV files
- We loop through the file to check if a next line is present. If so, then read next line, else break.

Simple text-based user interface

```
Welcome to Delfin SwimmingClub
Choose 1 to log in as trainer
Choose 2 to log in as cashier
Choose 3 to log in as Chairman
>
```

UI Logic

```
do {
    try {                                   //Try catch statement made by Nicolas with change sin logic when needed.
        System.out.print("Choose 1 to log in as Trainer\nChoose 2 to log in as Cashier\nChoose 3 to log in as Chairman\n>");
        whileKey =false;
        // Getting choice
        choice = sc.nextInt();
        sc.nextLine();
    }catch(InputMismatchException e){
        System.out.println("Not a legit answer");
        whileKey = true;
        sc.nextLine();
    }
}
// While choice is between 1 and 3 (the two boundaries included)
while (whileKey);
```

Trainer UI

```
Please enter Trainers name
Tobias
Please enter Trainers email
Tobias@gmail.com
Tobias Tobias@gmail.com
Enter crawl times
Enter swimmer 1's name

Fredrik
Enter swimmer 1's time
60
```

```
Welcome trainer
Enter number of swimmers
5
Choose swimming discipline.
Press 1 for crawl.
Press 2 for back crawl.
Press 3 for breast swimming.
Press 4 for butterfly times
```

```
Top 5 times:
Nicolas
45.0

Uldrik
56.0

Fredrik
60.0

Frank
78.0

Jeff
90.0
```

Cashier UI

```
Welcome Cashier
Please choose an option
1.      View Contingent
2.      Prompt Chairman to remove a member
3.      quit
Welcome Cashier
Please choose an option
1.      View Contingent
2.      Prompt Chairman to remove a member
3.      quit
1
Here is the total contigent
89500 kr per år

Please choose an option
1.      View Contingent
2.      Prompt Chairman to remove a member
3.      quit
2
Chairman has been prompted!
```

Chairman UI

```
Welcome Chairman
1 Remove member
2 Add member
3 Search data
4 Edit data
5 View contigent
6 View prompts
7 Exit program
Please write a number for the option you would like to choose
```