# INF112 - Compulsory assignment 3
# Institutt for informatikk
# Universitetet i Bergen

February 29, 2016

Please read all of the following carefully.

## Goals

- In compulsory assignments 3-5, the students will develop a software implementation of poker card games.
- The students will work on their implementations in teams according to their assigned groups.
- The first compulsory assignment includes a kick-off phase where students oranganize their teams and begin the project planning

## Group work organization

- During the development process, the students will actively exercise techniques of iterative development, SCRUM and the Unified Process (UP). The students will create corresponding artefacts during this work and these will be presented and evaluated.
- Each exercise corresponds to a sprint/iteration in an iterative development process, and consists of a number of exercises and implementation tasks. Completion of all the exercises ultimately leads to an implementation of poker.

## Grading and participation

- Each assignment will be graded seperately, with scores ranging from 0 to 100 points.
- Each group will be evaluated as a whole, based on work present in their respective repositories on the submission deadline, as well as a presentation that is to be held in class after each iteration.

- Students must attend at least 50% of the group meetings in order to be qualified to for examination for the course. Attendance at the presentations is mandatory.
- The group must get at least 40 points in each assingment in order for the students to be qualified for examination in the course.

## Practical information

- Project work must be organized in a git repository, where access is granted to all of the group members, the Teaching Assistant for the group and the lecturer.
- Only files in the repository will be evaluated for grading after each iteration.
- During each meeting, a student is to be assigned to write minutes from the meeting. Meeting minutes will give a brief overview of what was discussed, and should include an overview of tasks and to whom they have been assigned. The meeting minutes will be evaluated.
- In addition to the mandatory group meetings, we highly recommend each group organizes an additional weekly meeting on their own.

## Programming project, iteration 1 of 3

Each group will start developing a computer implementation of poker games from scratch. By the end of the iteration, the group will have develop the artefacts requested below.

**Submission deadline:** Wednesday, 16th of March, 2016, 20:00
**Project presentation:** Thursday, 17th of March, 2016, 10:15-12:00

### Project vision

#### Project overview

The aim of the project is to create a fully standalone computer version of a poker game following the official rules for ”texas holdem” poker. For a general overview of the game characteristics, see:
`http://www.wsop.com/poker-games/texas-holdem/`
and
`http://www.wsop.com/poker-games/texas-holdem/rules/`
for more specific rules.

The game will consist of an internal game logic that implements the rules and game mechanics and a graphical user interface. There will be different modes of interacting with the game either as game ”organizer/host”, ”spectator” or ”player”. In addition, the final version will include a multiplayer variant that allows to play against other human players via network connections.

*Sprites for the game cards can e.g. be downloaded here:*

`http://wheels-cards.wc.lt/bicycle_cards.html`

**Sprint 1**

The first version of the game should fullfill all requirements according to the following user stories:

- As an organizer, I want to initialize a poker table so that I can invite a group of players to join the game
- As an organizer, I want to invite a group of players to a poker table so that I can start a game
- As a player, I want to join a Sit & Go poker table for heads-up so that I can learn to play poker for fun
- As a player, I want to register for the games so that I can start playing poker and have fun
- As a player, I receive a set amount of chips when game starts so that I have something to wager
- As a player, I can see my cards, and the communal cards so that I can decide if to bet, check or fold
- As a player, I will be dealt no more cards if I do not have any chips left so that a winner can be declared
- As a player, I can see my position on the board, and the dealer-button so that I know what position i have (UTG, CO, BTN, SB, BB)
- As a player, I want be able to choose (basic) AI players as opponents so that I can play even if no other human players participate
- As a spectator I want to watch a poker table + player statistics so that I can learn strategies
- As a teaching assistant I want to see a tidy git repo so that I can easily find code, documents and models
- As a teaching assistant I want to find a README.md in the repository's top folder so that I can easily find licence information, group members and information on the project

## Exercise 1: Student introduction

First, assign a student to write minutes for the group meeting. This task will be rotated throughout the group meetings. Each student in the group will introduce themself, with name and academic background, as well as a brief overview of any programming experience. This will be written down by the student writing the minutes.

## Exercise 2: Git repository set-up

The Teaching Assistant for each group will create a corresponding git repository, and give access to all the students in the group, as well as the group Teaching Assistant and the course lecturer. Naming convention for the repositories: `INF112v16-G`*N* for group *N*. Each student will clone the repository. The student introduction notes will be uploaded to the git repository by the student writing minutes.

We want the repositories to contain the following directories (more can be added as necessary)

| *Folder name* | *Description* |
|---|---|
| `doc/` | Project documentation |
| `doc/minutes/` | Minutes from each group meeting |
| `src/` | Source code |

## Exercise 3: Project discussion

Begin discussing the project. This includes discussing:

- Getting an overview of the poker game (rules) and discussing the user stories for the first sprint.
- The general structure of the software implementation. Focus should be on the features requested in the exercise. However, it's beneficial to consider that the software will need to be extended in future iterations.
- The most appropriate assignment of task for the first iteration. It's important to note that the documentation produced (such as different diagrams) is of high importance for the evaluation of the group. In other words, please make sure to not only focus on the programming tasks.
- Any choices of libraries, such as for the Graphical User Interface for the game. These should be agreed upon and used by all group members throughout the project.
- Choice of a programming environment (IDE) and software for generating the required diagrams. Use of these should also be consistent among all group members.

Remember to note the decisions made in the meeting minutes.

## Exercise 4: Development

Implement the main classes of the poker game such that it fulfills the features requested in the project vision and create the deliverables given below accordingly. Also think about how you want to present the game functionality after the first sprint.

## Deliverables

Each group must submit the following on their group git repository by the submission deadline.

- User requirements specification
- Glossary (of non-trivial terms used in the software description)
- Use case diagram
- Use case text
- Domain model
- System sequence diagram (SSD) for the main success scenario, and for one alternative/exception.
- Operation contracts for three events from the SSD
- Source code and ANT build script (including a target called *JUnit*), preferably licensed with an *free software license*. [Alternative to ANT scripts, you may use other build frameworks that are free for use (e.g. `Maven`). If you do, all members of the group have to use the same program and clearly reference it in the minutes and project documentation.]
- Sprint backlog (user stories)
- Minutes from group meetings

The list of user requirements shall be formatted as follows:
`| ID | Description | Type | Priority |`
where `ID` is a unique identifier, `Description` is a description of the requirement, `Type` is either functional or non-functional and `Priority` is according to the MoSCoW method (`http://en.wikipedia.org/wiki/MoSCoW_method`).

## Formats accepted

Each diagram must be uploaded in a separate file. Diagrams generated from code will not be accepted. The following formats must be used for the submissions. Use of different formats will be penalized, such as with a reduction in the score for the corresponding deliverable.

- Diagrams: `png`, `jpg` or `pdf`
- Specifications: Text (`.txt`, *utf-8*) or `.pdf`
- Sprint backlog: Text (`.txt`, *utf-8*), `.odf` spreadsheet or `.pdf`
- Minutes: Text (`.txt`, *utf-8*)
- Source code and build script in text (*utf-8*)