TENTAMEN (EXAMINATION)

Tentamensdatum/*Examination date:* 17-08-24
(åå-mm-dd/*yy-mm-dd*)

	Ifylles av student	Ifylles av vakt
AID-nummer *AID number*	1495	1495
	Completed by student	*Completed by supervisor*

Kurskod/*Course code:* TDDC88 **Provkod/*Exam code:*** TEN1

Kursnamn/*Course title:* Programutvecklingsmetodik

Institution/*Department:* IDA

Inlämnat: antal blad 10 **provformulär** ☐
Enclosed: number of sheets ***exam booklet***

Markera behandlade uppgifter med X/*Mark tasks attempted with an X*

X här/*here*	1 X	2 X	3 X	4 X	5 X	6 X	7 X	8 X	9	10	11	12	13	14	15
Erhållna poäng *Points obtained*	9	10	8	7	7	8	20	4							
X här/*here*	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Erhållna poäng *Points obtained*															

Anvisningar/*Instructions*

1. Skriv AID-nummer, datum, kurskod och provkod på varje blad som lämnas in/
   Write AID number, date, course code and exam code on every sheet that is handed in
2. På varje papper får högst en uppgift lösas om inget annat anges/
   Maximum one task per sheet unless otherwise instructed
3. Skriv endast på papprets ena sida om inget annat anges/
   Use only one side of each sheet unless otherwise instructed
4. Numrera de papper som lämnas in/*Number every sheet that is handed in*
5. Använd inte röd penna/*Do not use a red pen/pencil*

Sen inlämning ☐
Late hand in

Klockslag _____
Time

Orsak _____
Reason

_____

∑ Poäng/*Points:* 81 (8) Betyg/*Grade:* 4

Examinator/*Examiner:* Ters

LINKÖPINGS UNIVERSITET

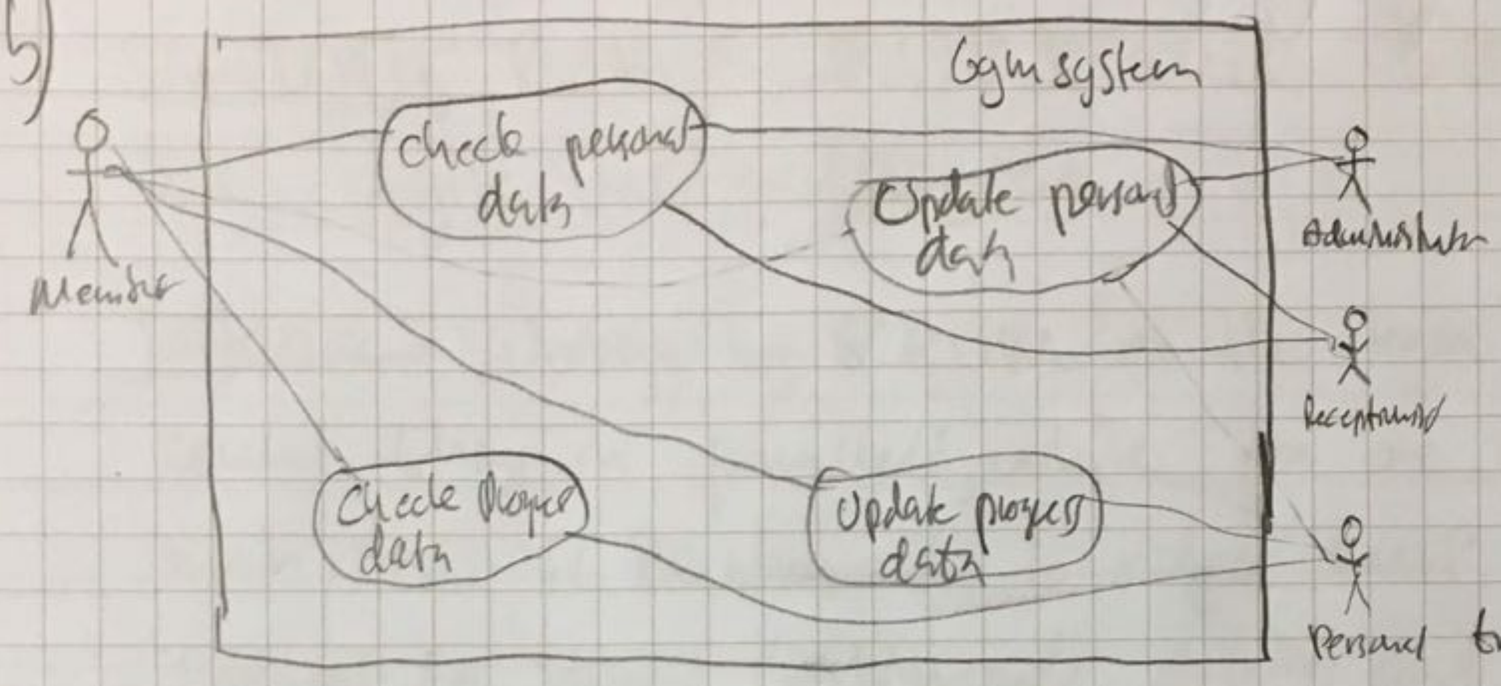
AID-nummer: *AID-number:* 1495	Datum: *Date:* 17-08-24	Blad nummer: *Sheet number:* 1
Kurskod: *Course code:* TDDC88	Provkod: *Exam code:* TEN1	

LINKÖPINGS UNIVERSITET

1. a) B, C

2

b)



Check personal data: The member or receptionist/administrator enter system to check a members personal data, eg:

- name
- address
- phone
- membership category
- ~ " — validity
- payment option

Update personal data: The administrator/receptionist enters system to update a members personal data. too short

Check progress data: A member or personal trainer enter System to check progress data (type of exercises, workload, number of repetitions, training record, personal goal) too short 3-

Update progress data: Same as above but update instead of check.

c) R1 Good: Use of modal verb "shall". +

Bad: Hard to test if req is fulfilled with unclear desired part "optimal". of an average customer registration +

R2: Good: Easy to test req. +

Bad: Could split this into two: +

- "The customer shall be ask to login with a FB-id;" +
- "The logged in customer shall be ask to listen to personal playlist" +

4/9-

LINKÖPINGS UNIVERSITET

AID-nummer: *AID-number:* 1495	Datum: *Date:* 17-08-24	Blad nummer: *Sheet number:* 2
Kurskod: *Course code:* TDDC88	Provkod: *Exam code:* TEN1	

2. a) B, D

2

b) Similarity: Both SCRUM and Kanban are agile methodologies. +

Difference: Working by SCRUM the team develops several things in parallell, while one of the main ideas of kanban is to reduce WIP. +

Proponents for scrum: Continuously develop and improve several parts of the system +

Proponents for kanban: Minimize multitasking, maximize throughput and enhance teamwork. It eliminates overplanning and can deliver sooner by sequencing. + 4

c) Roles: what different roles will be used in the project.
- Good to make a better understanding of who should be responsible for different parts of project.

Skills: What different skills are required for completing the project.
- Good for recruiting project team.

Training: what types of training / skill increase activities for the group will be carried out in project?
- Good to know what you, as a team member, can expect to learn from project. 4

Report&Communication: How is reporting done and how does the team communicate in the project?
- Good to do things consistently, so that every team member knows what communication platform to use. 10!

Examinator/*Examiner:*

Betyg/*Grade:*

LINKÖPINGS UNIVERSITET

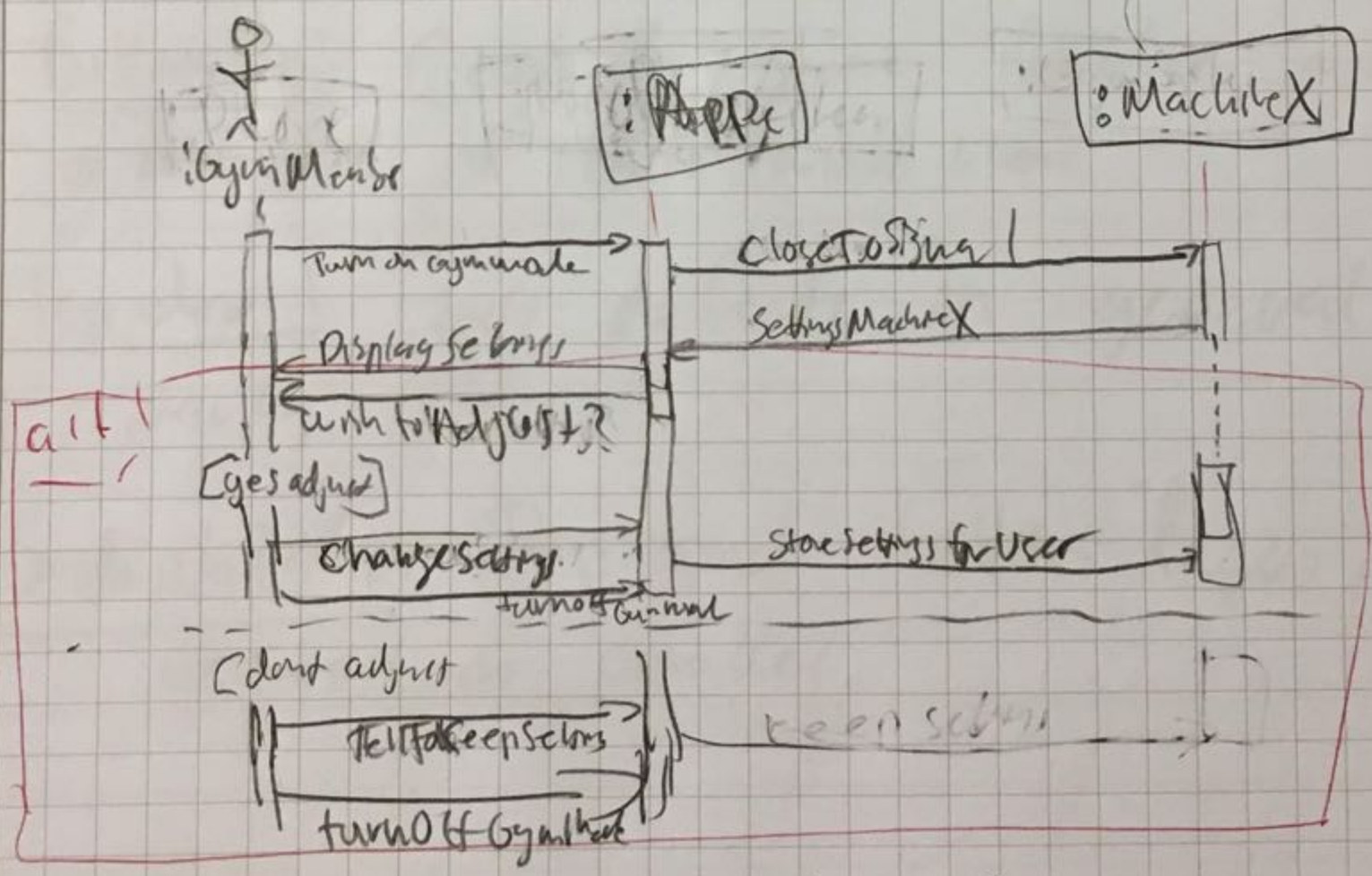
AID-nummer: *AID-number:* 1495	Datum: *Date:* 17-08-21	Blad nummer: *Sheet number:* 3
Kurskod: *Course code:* TDDC88	Provkod: *Exam code:* TEN1	

3a) C

b) • Easy to reuse layers. This is because less spaghetti are put into a layer which makes it more likely it could be reusal. This increases reusability since it makes it easier to reuse.

• Dependencies are kept local and modification is local to a layer. This is because dependencies are within the layer. This increases maintainability since with local dependencies it is easier to maintain system. 4

c)



3

/8

• Assumptions: only one machine (X)
• Settings stored IN machine (probably not optimal)

1

LINKÖPINGS UNIVERSITET

AID-nummer: *AID-number:* 1495	Datum: *Date:* 17-08-24	Blad nummer: *Sheet number:* 4
Kurskod: *Course code:* TDDC88	Provkod: *Exam code:* TEN1	

4

a) By 4

b) a) Developer pulls the latest commited version from the version handling tool.

1) Developer creates an own branch to do changes on (to not affect master branch until changes are done).

2) Developer does changes on branch.

3) Developer merges branch w. the master recently pulled

4) Developer pushes and commits master branch to the version handling tool, so that whole team gets changes.

c) Big Bang: System-wide testing on all levels means it will cover a huge part. (+)

Bottom Up: Good if basic functions are complicated or risky since it starts with them. (+)

Top-down: Finds defects in general designs early. +

Sandwich Testing: Top and bottom layers can be tested in parallel. +

4-
7-

LINKÖPINGS UNIVERSITET

AID-nummer: *AID-number:* 1495	Datum: *Date:* 17-08-21	Blad nummer: *Sheet number:* 5.
Kurskod: *Course code:* TDTC88	Provkod: *Exam code:* TEN1	

5. a) A, B

2

b) A specific area where you want to improve maturity. Let's say the backend team lacks most process maturity. Then team will set on going through the 5 CMMI steps (Initial → Repeatable → Defined → Managed → → Optimized) on that specific area. 1-

you mix staged and continuous models

c) 1) Inspection leader (moderator)

- Plans and organizes tasks
- Must be trained in inspection process
- Ensures inspection data is collected
- Issues inspection output

2) Recorder

- Documents defects, decisions, recommendations etc,
- Can be same as inspection leader.

3) Author

- Performs rework to meet inspection exit criteria
- Responsible for meeting entry criteria
- Shall not be inspection leader, reader or recorder

4) Reader

- Highlights important aspects
- Informs the software product to be inspected.

All of the above roles are also inspectors.

4

1/2-

Betyg/*Grade*

AID-nummer: *AID-number:* 1495	Datum: *Date:* 17-08-21	Blad nummer: *Sheet number:* 6
Kurskod: *Course code:* TDDC88	Provkod: *Exam code:* TEN1	

6a It doesn't make ~~as~~ much sense to do Boundary value testing on real numbers compared to integers, since we can't reach the edge cases in the same way. Therefore I would recommend more focus on other test methodologies, ~~both~~ ~~BVA~~ within Equivalence class testing.

However it might be possible that, even if the ~~edge~~ real edge cases are unreachable, there might be higher fail possibility when we are close to the boundaries. In that case I still think it might be worth to do the testing, where the test values come as close to the boundary as porible.

8

Faults?

Limited precision

7b

Rule 1

a) A standup meeting starts each day

b) A short standup meeting where each member walks through what he/she have done since last time, if any problems occured and what he/she plans to do next.

c) To always be up to date on everyone's process and to be able to tackle problems immediately. With 8 programmer doing a game it's important to always keep track on what is done. Makes it easier to avoid big setbacks if they don't come as surprise.

d) Time-consuming for little size and can irritate people if they aren't used to standups. 4

Rule 2

a) Code must be written to agreed standards

b) Every programmer shall write the code accordig to a standard agreed upon, eg how to name classes.

c) To be able to modify others code and to easy do adjustments same standards are important, since it's a game (in this example) it will probably be continuously changed and updated, and knowing what standards to use make it easier.

d) Risk that some of the programmers are used to work in other standards. 4

LINKÖPINGS UNIVERSITET

AID-nummer: *AID-number:* 1495	Datum: *Date:* 17-08-21	Blad nummer: *Sheet number:* 8
Kurskod: *Course code:* TDDC88	Provkod: *Exam code:* TEN1	

Rule 3

a) Refactor whenever and where ever possible

b) Re-writing and deleting code blocks that don't change result, but make code more readable.

c) Makes it easier to reuse code and to make changes later on, as well as to let other people change code. Important since changes probably will be made continously.

d) Takes time where development could be done instead. 4

Rule 4

a) Integrate often.

b) Integrate separately written code with the whole project often, instead of waiting and integrate many functionalities at same time.

c) Makes it easier to deal with smaller problems during the process instead of trying to deal with ~~bug~~ making huge problems. Reduces risk of putting effort into something that can't be integrated.

d) Make pauses from development. 4

LINKÖPINGS UNIVERSITET

AID-nummer: *AID-number:* 1495	Datum: *Date:* 17-08-21	Blad nummer: *Sheet number:* 9
Kurskod: *Course code:* TDDC88	Provkod: *Exam code:* TEN1	

Rule 5

a) Make frequent small releases

b) Release results frequently while still developing instead of making a final product.

c) Good in this game example, to release alpha and many beta versions to be able to receive customer feedback and improve game based on that until final release.

d) Risk of making "bad" releases so that customers won't like it and hence won't use final release.

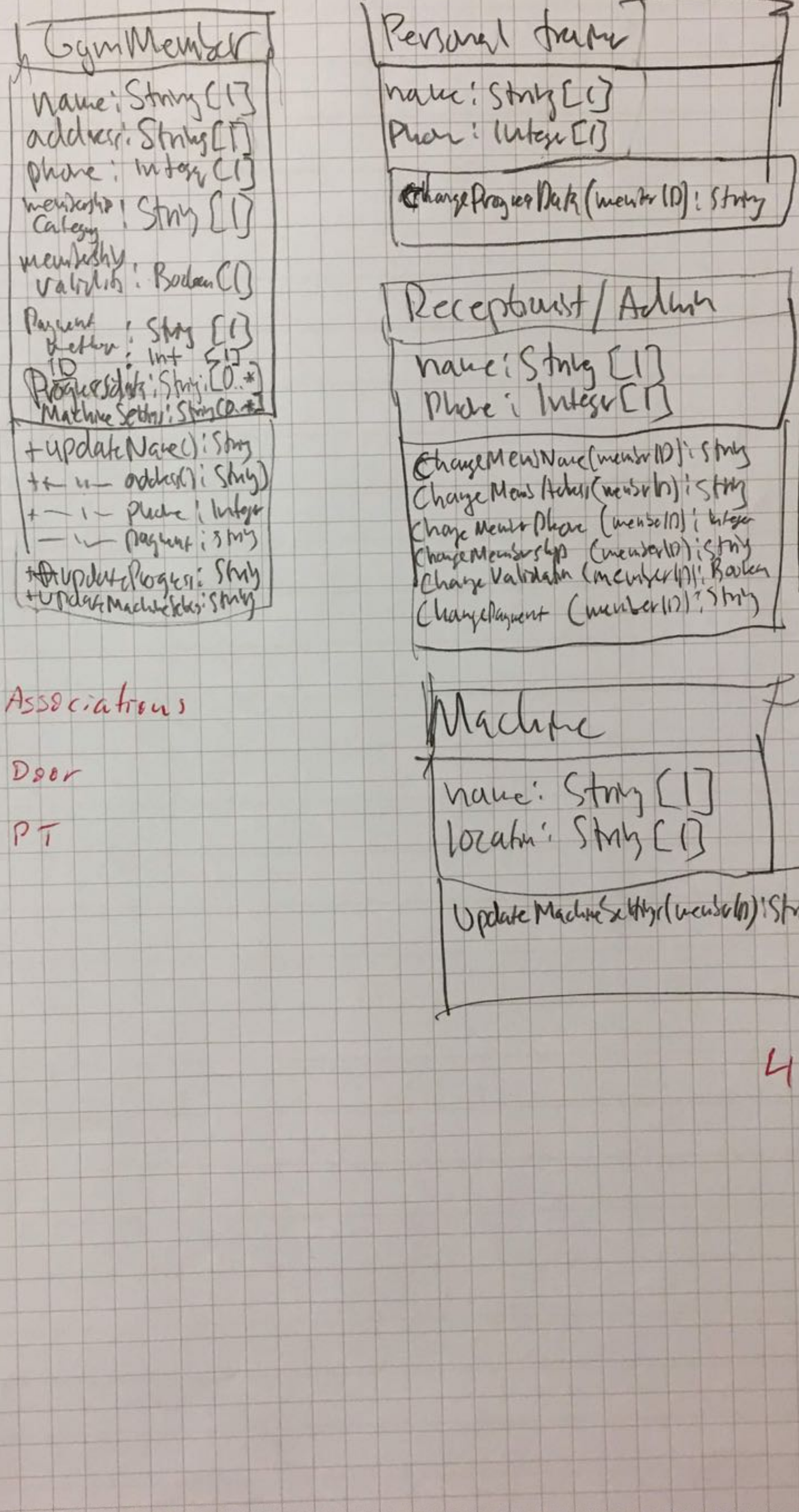
4-

20!

AID-nummer: *AID-number:* 1495	Datum: *Date:* 17-08-21	Blad nummer: *Sheet number:* 18
Kurskod: *Course code:* TDDC88	Provkod: *Exam code:* TEN1	

8.

b)



Associations

Door

PT