Brainstroming:

- How might we reduce the barrier to entry for people to start out with programming? (Reduce error messages, compile errors, setup of the environment, clear interface etc.)
    - (lu) build our own environment where no error messages and such problems are possible.
        - (Ch) we could package all elements into a very high-level block based interface so that to escape from some, for example ,syntax errors
        - (Ch) or maybe we could explain or clarify the error in a more readable expression, so that the user could also gain some knowledge of troubles/error message might have in other systems
            - (ms) create some example solution that can be consulted by the user when he is stuck.
    - (lu) App where the user specify what project they want to do and we recommend them the tools and languages they should use / learn
        - (ms) give the user hints in the task how he should start and what could be missing from his implementation.
    - (lu) build an IDE like environment that teaches the user a language they select
    - (ms) coding in browser
    - (ms) use speech-to-code tools to give users a feel for how the linguistic thought process translates to programming syntax

- How might we help people to figure out what programming skills they should learn to match their needs?
    - (ms) Questionaire with different categories on skills (e.g. databases, algorithms, datastructures, control flow) to see what skills they already have and then we fill out the missing gaps
        - (lu) focus questionnaire on what their needs are and show skills needed afterwards
        - (Ch) for users who have not that clear idea(e.g. what they want/need or what programming fields could there be), should be first wide and then narrow down the scope
        - (Ch) should be able to divide users of different level
    - (ms) compilation of skills required for different jobs or simply a list where they can choose which skills or lessons they would like to learn.
    - (ms) Have an abstraction of skills that people with no experience can understand. They might want to do automation of daily tasks -> we then offer basics of scripting.
        - (lu) make it possible to increase difficulty / more advance stuff

Human Computer Interaction Hand-in 1

- How might we introduce fundamental concepts to beginners and guide them through the learning process (difference between children and adults)? (abstraction of content)
    - (lu) Self study and practice guided by class like tutorials
        - (ms) self study might need some text. Either we write that text/introduction ourselves or we link some useful learning resources (wiki, videos etc.)
    - (lu) Interactive visual representations of the concepts (gamified)
        - (ms) something like a FAQ or a forum where participants can ask questions and can see thing that confused previous participants
        - (Ch) may have some dynamic hint/suggestions and user could choose to see
    - (lu) Video tutorials
    - (ms) optional tutor that can be contacted in case there is a problem
        - (Ch) could automatically start when a user start using a new element for the first time, and user could choose to close it/enter it later just like the first time u r using some other tool/playing some game
    - (ms) show ranking of difficulty of lesson
        - (Ch) some harder course may be unlocked/suggested to take after user finish some previous part or finishing some test

- How might we deepen or refresh the skills of already experienced programmers?
    - (ms) for a refresher we first need to know what they already have an understanding what the user already knows. They should be able to choose from a list what they want to do. Since they are experienced they know what kind of topics should be available
        - have the option to let the user test themselves, in order to figure out how good their understanding of a topic really is
        - (Ch) have the tutorial/task that could optionally skipped
    - (ms) there should be multiple degrees of how sophisticated the topic is explained. (beginner, intermediate, expert, “I basically invented this stuff”/God-mode)
        - (lu) also present them challenges to solve in their respective topic and learn hands on
    - (ms) to deepen the understanding we also need to provide literature recommendations or recent research (published papers). Especially useful in topics like machine learning if you want to improve your models
        - (lu) but make sure the literature is also friendly to intermediates
    - (Ch) could have a recommendation system to recommend more advanced topics (e.g. algo/data structures), or proper challenging big project for the user, to give them more ideas on what they could learn/practice further

- How might we communicate to people that programming is beneficial for their career?
    - (Ch) may suggest some job opportunities after the user have equipped some skills/finished some tutorials

        - (ms) work together with companies s.t. users can directly go to websites with job openings
            - (lu) have the option for companies to present themselves in events
    - (Ch) may have some functions/programming tasks related to some certain background / solving some real world problem
        - (ms) show them the comparison in terms of workload before and after automation of a task.
            - (lu) show them concrete examples on how they can automate a daily task
    - (Ch) may “grant” some “certificate” suggesting users level of comprehensive, which may have reach the level equivalent to some standard of education
        - (ms) collaboration with universities s.t. the certificate is approved by some institute
            - (lu) work with companies to grant them internships
    - (ms) create a newsletter where we advertise our courses
    - (Ch) could have some course/tutorial aimed at gaining skill for getting some specific jobs

- How might we make learning programming fun/not boring and challenging? (presentation of content)
    - (Ch) Setting small tasks / games (from easy to hard) where users could possibly gain some “awards”
        - (lu) make a “score” system such that friends (or strangers) an compete against each other
        - (ms) create time challenges after a lesson is complete (speedrunners welcome)
    - (Ch) Design functions that user could see the effect of the program/change of code immediately (visualized result?)
        - (lu) make it possible to show different functions at the same time so that users can compare the functions to each other
            - (ms) option to split coding environment into multiple windows
            - (ms) do we need to teach people how to use the terminal?
    - (Ch) make the interface more interesting/interactive
        - (lu) make the interface intuitive and easy to use
        - (ms) bright colors for children, memes for teenagers, possible salary for adults :)
    - (ms) when someone goes through the task very quickly then the following tasks become optional/ are skipped