ATM Design

In this assignment you will practice the steps involved in **Problem Analysis** and **Design** stages of the Software Development Process. You are to research and design a program that simulates an ATM machine. ***You will not actually be writing the code for this program.***

Analysis - *not submitted*

1. Gather information about ATMs. You may use the following video, or search for other sites with information on how to use bank machines online.

    - YouTube video on how to use an ATM

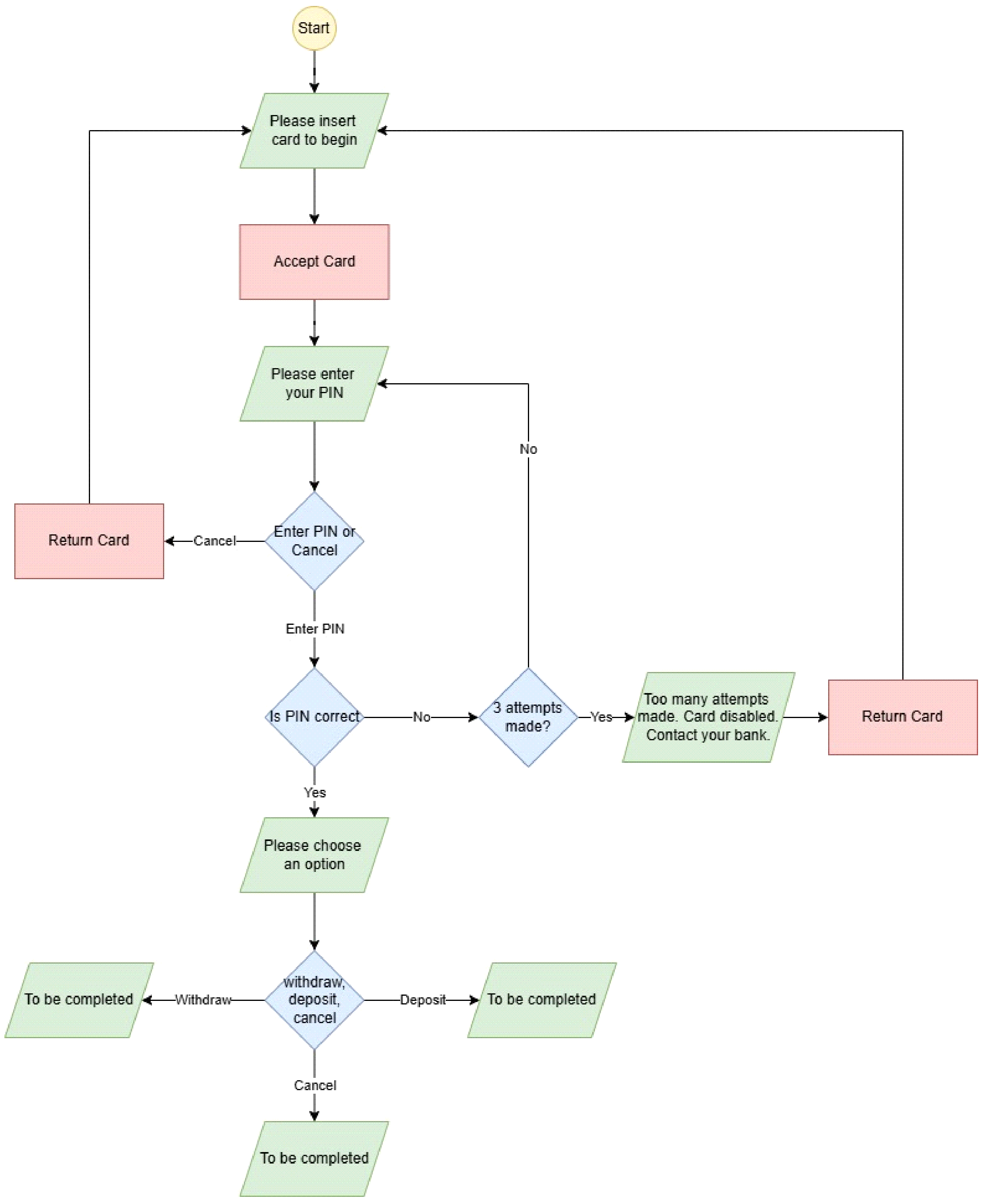
Not all ATM's work the same way. For this assignment only consider the following operations: **Deposit,** and **Withdrawal**.

Design - *flowchart (to be submitted)*

For each of the following processes create a flowchart, (**draw.io**), that shows how they work. Be specific, clear, and thorough.

1. PIN entry
    - This is already done for you.
    - Use the flowchart at the end of this section that shows the PIN entry process as a starting point
    - The next 2 processes should be added to that flowchart
2. Withdrawal
3. Deposit

For each transaction type consider the following:
- The user needs the ability to cancel out of a transaction.
- For our purposes assume each user has two accounts, Savings and Chequings.
- A user should have the option of a printed receipt when a transaction is complete.



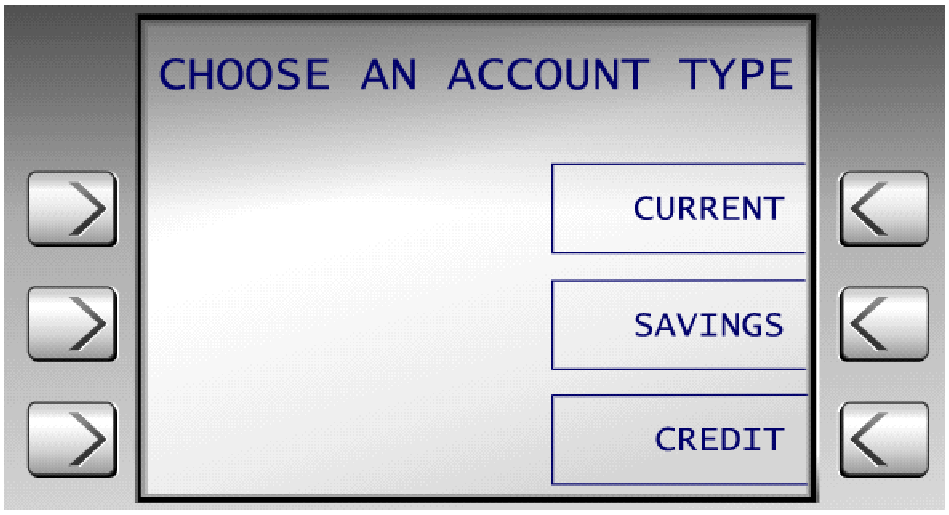
HINT: Think about what happens in your design when the user generates an error. For example if the user tries to take out more money than they have.

Design - *mockups (to be submitted)*

Use draw.io to create layouts of **all** the screens in your program. They do not have to be extremely detailed, however they should clearly show all the inputs and outputs of the screen, and links are to be created that go from one screen to the next.

Export your work as an html file for submission.

Below is an example of a screen showing the selection of an account during a withdrawal process:



Categories	**Level 1 (50 - 59%)**	**Level 2 (60 - 69%)**	**Level 3 (70 - 79%)**	**Level 4 (80 - 100%)**
Algorithm Creation develop appropriate algorithms in text or diagram form to solve problems and verify solutions	- demonstrates limited ability to develop appropriate algorithms in text or diagram form to solve problems and verify solutions	- demonstrates some ability to develop appropriate algorithms in text or diagram form to solve problems and verify solutions	- demonstrates considerable ability to develop appropriate algorithms in text or diagram form to solve problems and verify solutions	- demonstrates a high level of ability to develop appropriate algorithms in text or diagram form to solve problems and verify solutions
Screen Layouts design user-friendly software interfaces (e.g., prompts, messages, screens, forms);	- design is confusing, and contains missing elements (e.g. spelling, alignment , poor colour choices, no ability to enter PIN, etc.)	- all elements present however some errors still persist, (e.g. spelling, colour, alignment, etc.)	- all elements present and clearly outlined, however minor problems still exist, (e.g. no screen linking)	- screens are well designed, easy to understand and accurately depict an ATM