Q1 [30 marks] ADT Design

You are to complete the implementation of a family of social network account ADTs.

You will be implementing several ADTs as part of your solution to Questions 2 and 3 of this assignment. That question is about finding connections between users of various social networks.

In this assignment, you’ll be defining the following ADTs:

- **EmailAddress** represents a unique string of the form `local-part@domain`. To keep things simple, we'll restrict the email address to a total of 255 characters, including the '@' symbol, and the strings to alphanumeric characters plus the period ('.') and the hyphen ('-'). There are no restrictions on the quantity of periods or hyphens that may occur in either string, and the only restriction on placement of the period or hyphen is that it may not be the first character i.e. the first character of both the `local-part` and the `domain` must be alphanumeric. Neither the `local-part` nor the `domain` may consist of an empty string.

- **User** represents a person who may participate in 0 or more social networks, and thus have an account in each network that will be associated with a user via the unique `EmailAddress`. Users also have a name (represented by a string), which may not necessarily be unique.

- **Collection** represents a set of unique `User` accounts in program. There may be more than one collection, and their contents may overlap.

For each of the above ADTs, answer the following design questions.

a) State whether the ADT is an entity- or value-based ADT, and explain your rationale for this decision.

b) Explain how whether the ADT is an entity- or value-based ADT affects its design and implementation (e.g., should the copy constructor be allowed? Should it implement a deep or shallow copy?).

c) For each of the five C++ special member functions, state whether you would implement your own or whether you would use the compiler-generated version if applicable. Briefly explain your rationale for each decision.

Note that the five special member functions for the purposes of this question consist of: the default constructor, the destructor, the copy constructor, the assignment operator, and the equality operator. We are treating the "move" versions of the constructor and assignment operators as special cases of the copy constructor and assignment operator and thus not discussing them separately.

---

1 Described by the `addr-spec` field defined in [RFC 5322](https://tools.ietf.org/html/rfc5322) if you want to know the details.