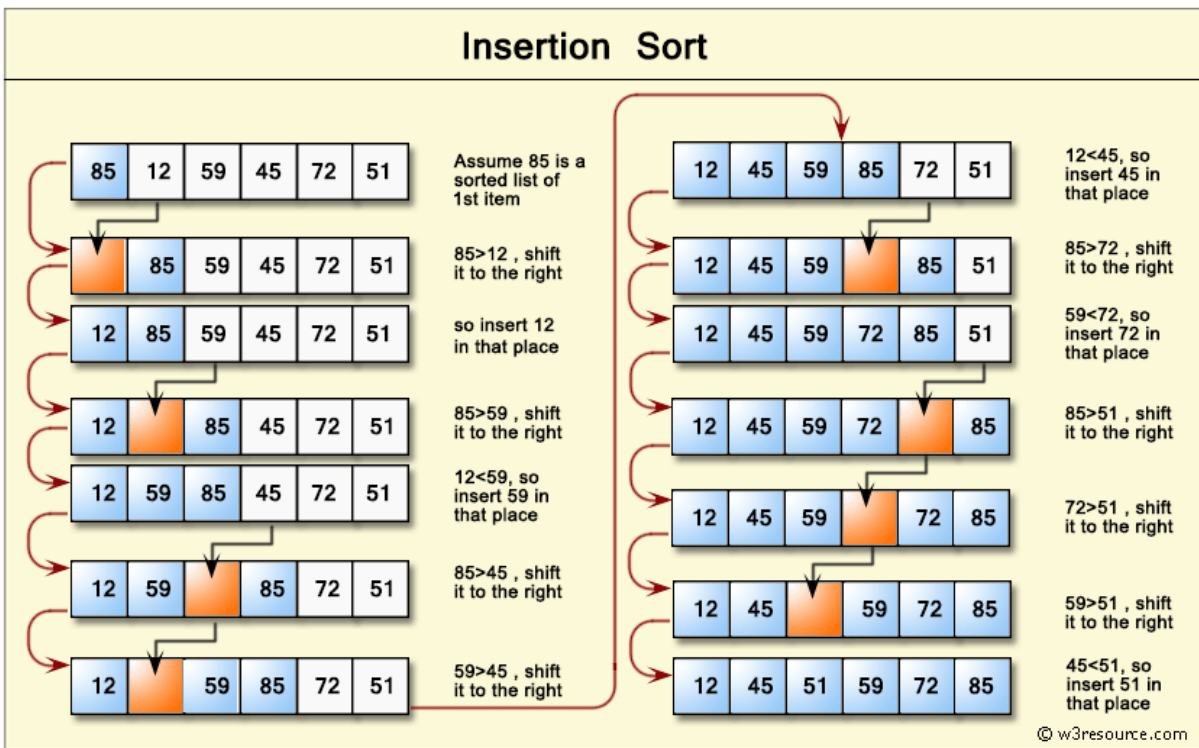


1. Sorting

Create an array of 10 random numbers (0,1000) and try insertion sort. Do it step-by-step and yourself and try to understand. Add short comments after every line of code explaining what is the purpose and what is done there.

Insertion sort is a comparison-based algorithm that builds a final sorted array one element at a time. It iterates through an input array and removes one element per iteration, finds the place the element belongs in the array, and then places it there.



Example: https://www.w3resource.com/w3r_images/Insertion-sort-animation.gif

Tips: use smaller list in the beginning; print out the results after every step.

2. String manipulation

Create a system for creating and decoding user accounts and names for a domain.

Rules are:

1. Username:

- Username is created by putting "." sign between forename (*eesnimi*) and last name (*perenimi*):
"Juhan Juurikas" -> *juhan.juurikas*
- If user has more than one forename then they are joined:
"juhan mart juurikas" -> *juhanmart.juurikas*
- Username has to always be in lowercase letters
- Username can consist only of one last name (and thereby one "." sign)
- Username cannot contain ""äüö" (replace them with a and o)

2. Email account:

- Account name consists of: username, domain and .eu
juhan.juurikas@company.eu

3. Domain account

- Domain account consists of: domain, "/", username
company/juhan.juurikas

Create methods for:

- 1) Finding user name and domain based on email account and domain account.
FindUserName("juhan.juurikas@company.eu") -> Name: Juhan Juurikas Domain: Company
FindUserName("company/juhan.juurikas") -> Name: Juhan Juurikas Domain: Company
- 2) For generating email account based on name(s) and domain:
CreateEmailAccount("Juhan Juurikas company") -> juhan.juurikas@company.eu
CreateEmailAccount("Juhan Märt juurikas company") -> juhanmart.juurikas@company.eu
- 3) For generating domain account based on name(s) and domain:
CreateDomainAccount("Juhan Juurikas company") -> company/juhan.juurikas

All methods take one string type parameter and they also return one string value! (You need return values for testing!)

Add NUnit tests for all methods! (At least 2 tests for every method) Think how you can test FindUserName method (the structure of the return value).

Username is needed for both domain account and email account: make it as a separate method. Avoid code duplication. Think of the structure: what is fixed (for example always one domain) and what can change (there can be multiple forenames). Use Console.WriteLine() to check your values.

Tips:

<https://www.dotnetperls.com/tolower>

<https://www.dotnetperls.com/indexof>

<https://www.dotnetperls.com/split>

<https://www.dotnetperls.com/substring>

<https://www.dotnetperls.com/replace>

