DVA245, Laboration 1 – Array insert at

Mål
1- kunna använda abstrakta datatyper i programmeringsuppgifter
6 - vara tillräckligt bekant med några specifika abstrakta datatyper för att vid behov kunna lägga till operationer på dessa. Exempel på sådana abstrakta datatyper är binära träd, dynamiska listor, direktacesslistor, olika sökdatastrukturer, grafer

Utöka klassen `Array` i filen `Array.py` med en medlemsfunktion `insert_at` som tar två argument, `position` och `item` utöver `self`. Funktionen ska sätta in värdet `item` på indexet `position`.

Alla element/ värden som fanns i listan innan ska finnas även efter insättningen – men alla från `position` och till listans slut ska flyttas ett steg ”uppåt” – mot högre index.

Skriv tester för funktionen, i `ArrayClientLab1.py`, i en `main`-funktion i `ArrayLab1.py` eller i en separat fil. Använd ett `Array`-objekt där ni först satt in några värden med vanliga `insert`, innan ni testar funktionen. Testa att göra `insert_at` på positionen 0, i slutet av listan och på en plats där det finns värden både före och efter positionen, och kontrollera efter varje insättning att alla andra värden finns kvar i er lista på rätt platser, och att storleken (len) ökats på med 1.

Ni kan titta på funktionen för delete, där flyttas värden ett steg ”nedåt” för att fylla upp tomrummet i listan när ett värde tagits bort. Ni ska göra tvärtom innan ni kan sätta in det nya värdet på sin plats. Tänk på att ni måste börja med att flytta det sista värdet för att ha utrymme för det näst sista och så vidare. Pythons range har ett step-argument som du kan sätta till -1 för att stega nedåt från ett högt start till ett lågt stop.

https://docs.python.org/3/library/functions.html#func-range

För redovisning:

Ni ska kunna förklara hur testerna visar att er funktion fungerar, och hur funktionen fungerar.

English:

Learning outcomes
1- be able to use abstract data types in programming assignments
6 - be sufficiently familiar with some specific abstract data types in order to be able to add new operations on theses. Some examples of such abstract data types are dynamic lists, direct access lists various seach data structures, graphs

Extend the `Array` class in the file `Array.py` with a member function `insert_at` that takes two arguments, `position` and `item` in addition to `self`. The function shall insert the value `item` at the `position`.

All elements/ values that were in the array prior to the insertion shall still be in the array – but all from `position` to the end of the list shall move one step "upwards".

Write tests for the function, in a `main`-function or in a separate file. Use an `Array`-object where you inserted some values with the normal `insert`, before you test the function. Test to do `insert_at` at index 0, at the end of the array and in a position where there are values before and after the position, and make sure it works as intended.

You can look at the function for delete, where values are moved one step "downward" to remove the empty space when a value has been removed. You shall do the opposite before you can insert the new value at the right position. Note that you need to move the last value first to have room for the value before the last and so on. Python's range has a step argument that you can set to -1 to step downwards from a high start to a low stop.

https://docs.python.org/3/library/functions.html#func-range

For presentation:

You need to be able to explain how the tests show that your function works, and how the function works.