IB111 Úvod do programovaní skrze Python

Autor: Slavomír Krupa,
Text inšpirovaný Valdemarom Švábenským


Odpovedníky


SEMESTRÁLKA


ZÁSTUP

Radenie

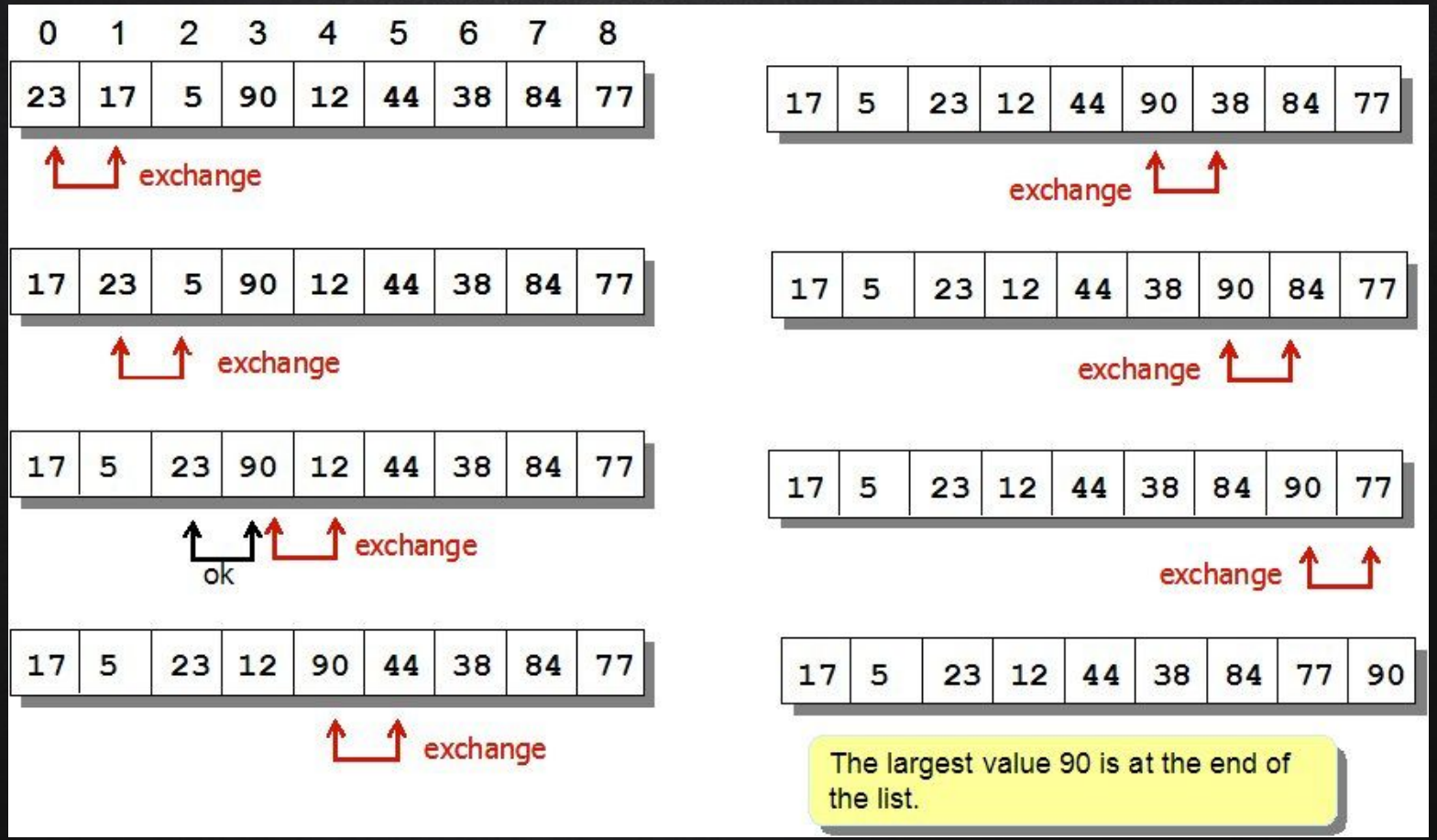
General

- ★ Vstup: zoznam hodnôt, na ktorých existuje usporiadanie
- ★ Výstup: usporiadaný zoznam hodnôt vstupného zoznamu
- ★ Rozdelenie zoznamu na dve pomyselné časti:
  - neusporiadanú časť a
  - usporiadanú časť.

- ★ `lst.sort()` a `sorted(lst)` sú implementované algoritmami nad rámec predmetu.

Bubble sort

- ★ Po každej iterácii najväčší prvok “prebuble” na koniec zoznamu
- ★ Algoritmus prechádza zoznam od začiatku a porovná každú dvojicu susedných prvkov
  - Ak sú susedia v opačnom poradí, vymení ich
  - Ak sú susedia v správnom poradí, nechá ich tak
- ★ Ak nastala aspoň jedna výmena, zoznam sa bude prechádzať znova od začiatku



Zdroj: http://www.csit.parkland.edu/~mbrandyberry/CS1Java/Lessons/Lesson28/BubbleSort.htm

Bubble sort

- https://youtu.be/lyZQPjUT5B4?t=52
- https://www.toptal.com/developers/sorting-algorithms/bubble-sort
- http://www.algolist.net/Algorithms/Sorting/Bubble_sort

Úloha 1

- ★ Naprogramuje funkciu `bubble_sort(lst)`, kde `lst` je zoznam hodnôt
- ★ Využite testovacie výpisy pred každou iteráciou cyklu pre kontrolu priebehu radenia
- ★ Môžete meniť vstupný zoznam

```
>>> bubble_sort([9, 8, 4, 6, 1])
[9, 8, 4, 6, 1]
[8, 4, 6, 1, 9]
[4, 6, 1, 8, 9]
[4, 1, 6, 8, 9]
[1, 4, 6, 8, 9]
[1, 4, 6, 8, 9]
Result: [1, 4, 6, 8, 9]
```

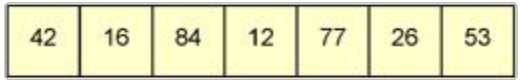
Doplnok úlohy 1

Upravte funkciu bubble sort `bubble_sort(lst)` tak aby sa radenie ukončilo keď nenastala výmena

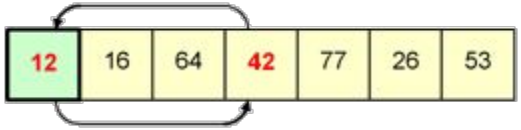
```
>>> bubble_sort([9, 8, 4, 6, 1])
[9, 8, 4, 6, 1]
[8, 4, 6, 1, 9]
[4, 6, 1, 8, 9]
[4, 1, 6, 8, 9]
[1, 4, 6, 8, 9]
Result: [1, 4, 6, 8, 9]
```

Selection sort

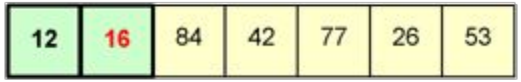
- ★ Najprv je zoradená časť prázdna a nezoradená časť je celý vstupný zoznam
- ★ V každom kroku algoritmus nájde minimum nezoradenej časti a vymení ho s najľavejším prvkom nezoradenej časti
- ★ Tým ho vlastne pridá na koniec (napravo) zoradenej časti
- ★ Zoradená časť sa buduje zľava doprava
- ★ Keď je nezoradená časť prázdna, algoritmus končí



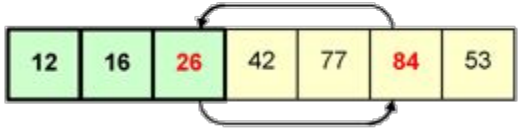
The array, before the selection sort operation begins.



The smallest number (**12**) is swapped into the first element in the structure.

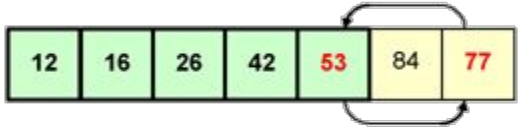


In the data that remains, **16** is the smallest; and it does not need to be moved.

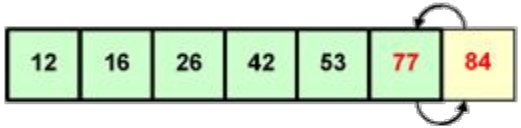


26 is the next smallest number, and it is swapped into the third position.

42 is the next smallest number; it is already in the correct position.



53 is the smallest number in the data that remains; and it is swapped to the appropriate position.



Of the two remaining data items, **77** is the smaller; the items are swapped. *The selection sort is now complete.*

Zdroj: http://www.cs.rmit.edu.au/online/blackboard/chapter/05/documents/contribute/chapter/09/array-sorting.html

Selection sort

- ★ https://www.toptal.com/developers/sorting-algorithms/selection-sort
- ★ http://www.algolist.net/Algorithms/Sorting/Selection_sort
- ★ https://www.youtube.com/watch?v=Ns4TPTC8whw

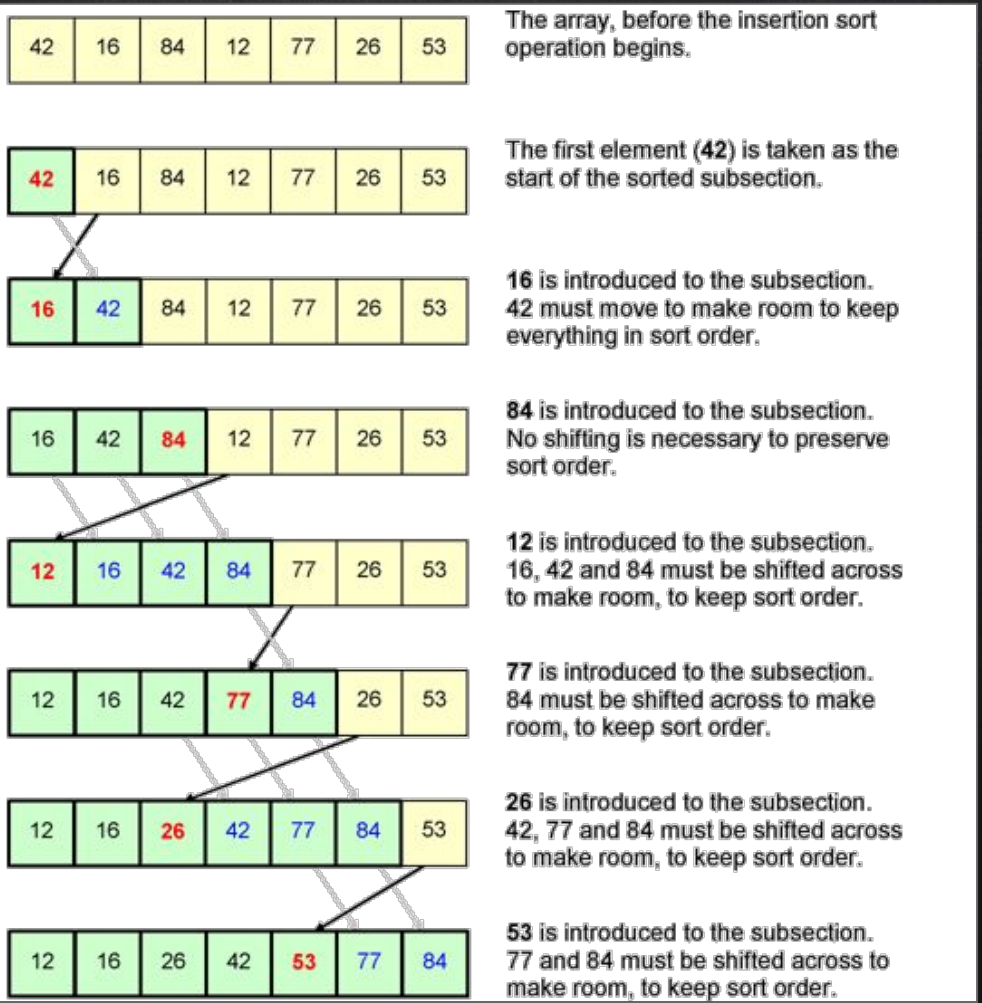
Úloha 2

- ★ Naprogramuje funkciu `selection_sort(lst)`, kde `lst` je zoznam hodnôt
- ★ Využite testovacie výpisy pred každou iteráciou cyklu pre kontrolu priebehu radenia
- ★ Môžete meniť vstupný zoznam

```
>>> selection_sort([9, 8, 4, 6, 1])
[9, 8, 4, 6, 1]
[1, 8, 4, 6, 9]
[1, 4, 8, 6, 9]
[1, 4, 6, 8, 9]
[1, 4, 6, 8, 9]
Result: [1, 4, 6, 8, 9]
```

Insert sort

- ★ Najprv zoradená časť obsahuje prvý prvok vstupného zoznamu a nezoradená časť je zvyšok zoznamu
- ★ V každom kroku algoritmus vezme prvý prvok nezoradenej časti a vloží ho na správne miesto zoradenej časti
- ★ Keď je nezoradená časť prázdna, algoritmus končí



Zdroj: http://www.cs.rmit.edu.au/online/blackboard/chapter/05/documents/contribute/chapter/09/array-sorting.html

Insertion sort

- ★ https://www.toptal.com/developers/sorting-algorithms/insertion-sort
- ★ http://www.algolist.net/Algorithms/Sorting/Insertion_sort
- ★ https://www.youtube.com/watch?v=ROalU379l3U

Úloha 3

- ★ Naprogramuje funkciu `insertion_sort(lst)`, kde `lst` je zoznam hodnôt
- ★ Využite testovacie výpisy pred každou iteráciou cyklu pre kontrolu priebehu radenia
- ★ Môžete meniť vstupný zoznam

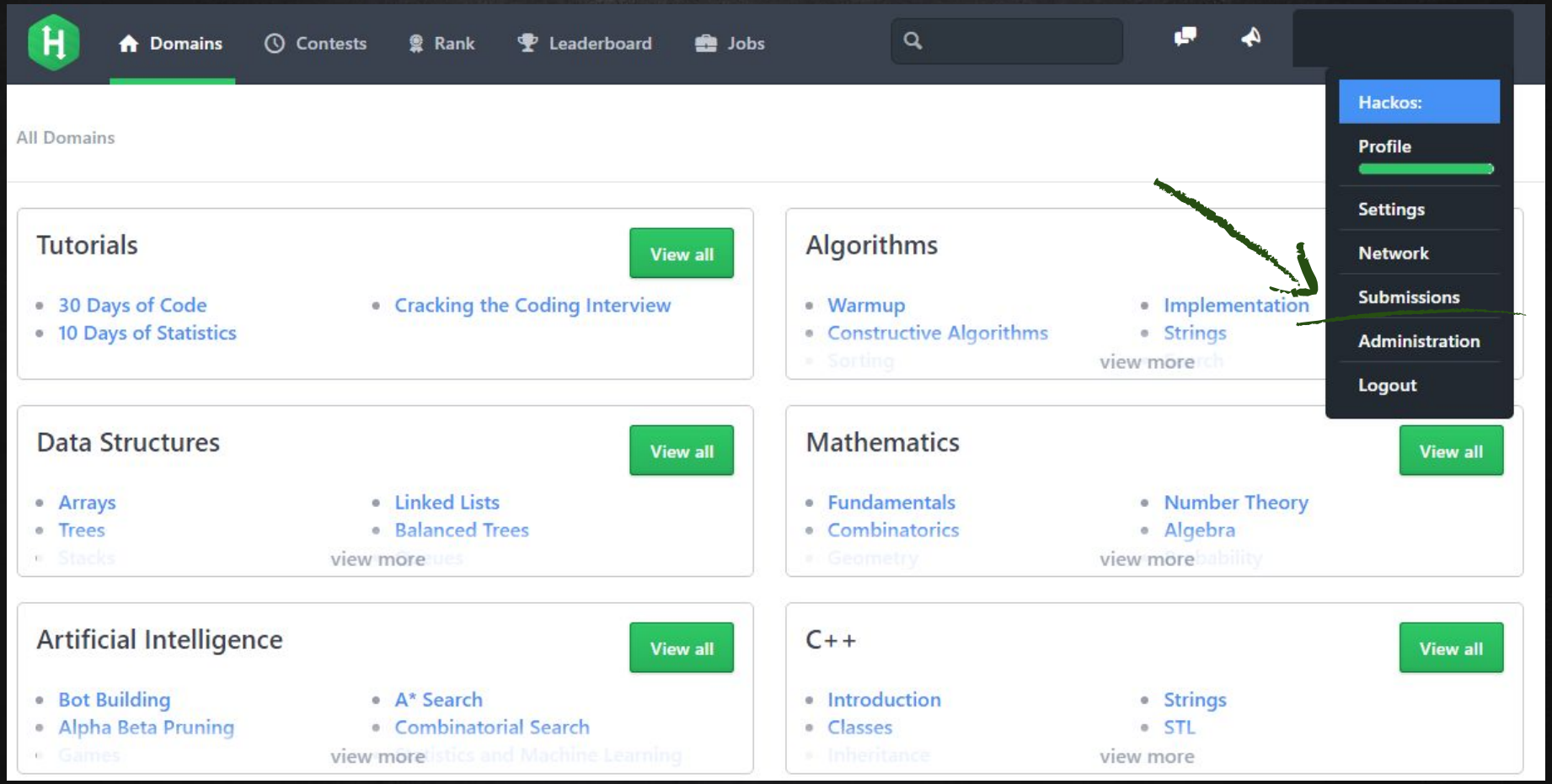
```
>>> insertion_sort([9, 8, 4, 6, 1])
[9, 8, 4, 6, 1]
[8, 9, 4, 6, 1]
[4, 8, 9, 6, 1]
[4, 6, 8, 9, 1]
[1, 4, 6, 8, 9]
Result: [1, 4, 6, 8, 9]
```

Doplnok úlohy 3

Upravte funkciu `insertion_sort(lst)` z predošlej úlohy tak, že použijete upravený binary search pre nájdenie pozície kde sa bude vkladať číslo.

?Pair programming?

DU


Domains
Contests
Rank
Leaderboard
Jobs
Hackos:
Profile
Settings
Network
Submissions
Administration
Logout
All Domains
Tutorials
View all
30 Days of Code
10 Days of Statistics
Cracking the Coding Interview
Algorithms
Warmup
Constructive Algorithms
Implementation
Strings
view more
Data Structures
View all
Arrays
Trees
Linked Lists
Balanced Trees
view more
Mathematics
View all
Fundamentals
Combinatorics
Number Theory
Algebra
view more
Artificial Intelligence
View all
Bot Building
Alpha Beta Pruning
A* Search
Combinatorial Search
view more
C++
View all
Introduction
Classes
Strings
STL
view more

Úlohy*

- ★ https://www.fi.muni.cz/IB111/sbirka/07-seznamy_algoritmy.html
- ★ `7.1.2`
- ★ `7.1.3`

Credits

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Unsplash