





Ефективност и избор на структурата данни

- 
- Основното послание на тази лекция е, че **ефективността на дадена програма зависи от избора на структурата данни.**

Ще демонстрираме това на примера на класа **EditorBuffer**, реализиращ **прост текстов редактор**.

Ще видим как различните представяния за буфера дават различна ефективност на отделните операции.



Изпращането на текстови съобщения е най-разпространената форма на комуникация в съвременния свят. Съобщения можем да изпращаме например с телефоните си, които поддържат текстов редактор.

Съвременните телефони (и не само те) използват в своя дизайн така наречения **Model-View-Controller pattern (MCV)**

Клавиатурата представя controller-а.

Дисплеят представя това което виждаме (view).

Структурата данни, която се използва за съобщението, представя модела.

Съобщение е редица от символи (наричана често буфер),
Имаме и мигаща вертикална черта (курсор), при която можем да добавяме и приемем символи.

Макар програмирането на дисплея и клавиатурата да е също така предизвикателно, ние ще се концентрираме върху програмирането на буфера на текстовия редактор.

FIGURE 13-1 Cell phone decomposition using the model-view-controller pattern

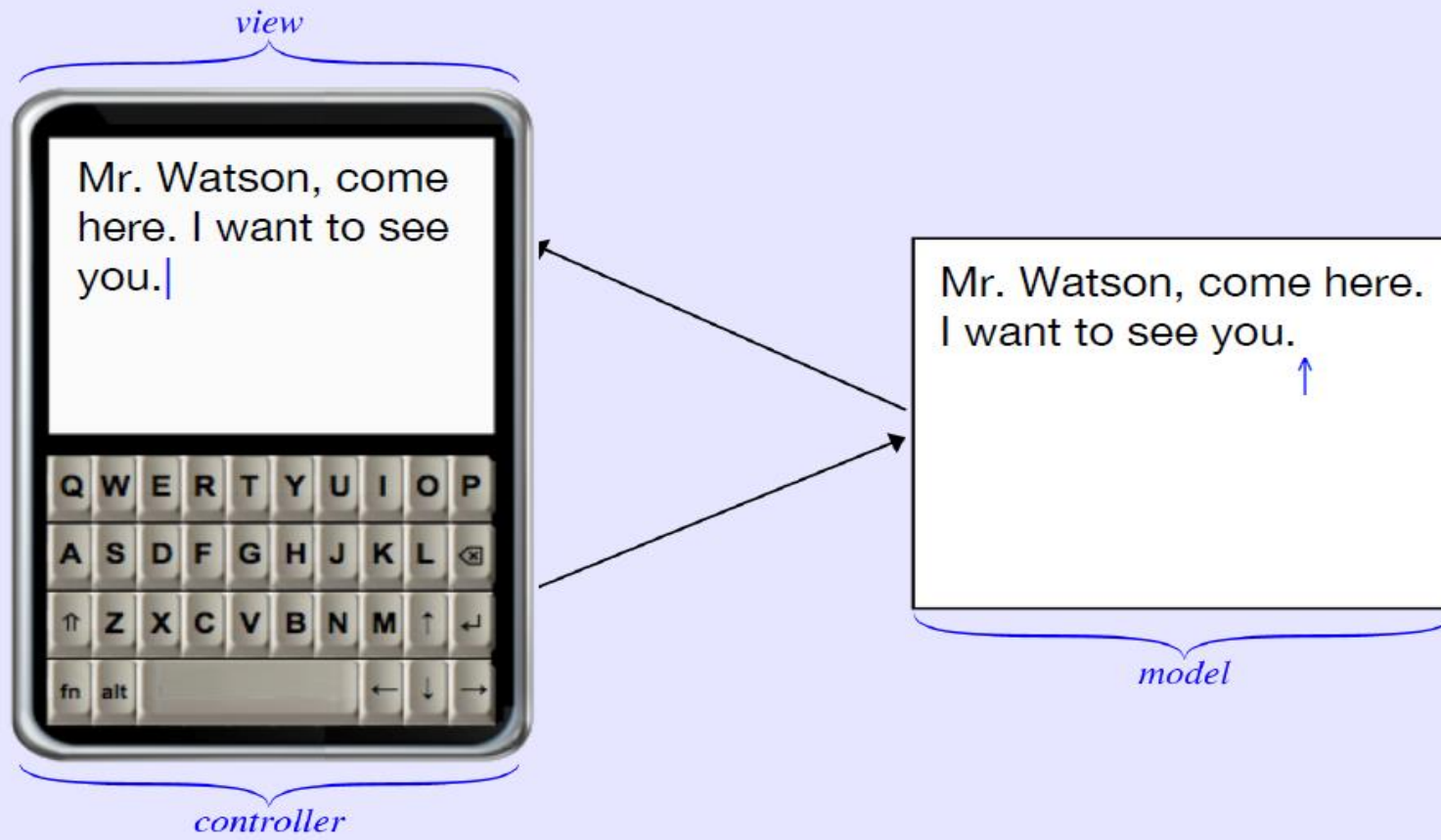


TABLE 13-1 Commands available in a simple command-based editor

F	Moves the editing cursor forward one character position.
B	Moves the editing cursor backward one character position.
J	Jumps to the beginning of the buffer.
E	Moves the cursor to the end of the buffer.
I_{xxx}	Inserts the characters <i>xxx</i> at the current cursor position.
D	Deletes the character just after the current cursor position.
H	Prints out a help message listing the commands.
Q	Quit the editor program.

Когато нашата програма се изпълнява,
искаме екранът да изглежда така:

```
*Iaxc
  a x c
      ^

*J
  a x c
  ^

*F
  a x c
    ^

*D
  a c
    ^

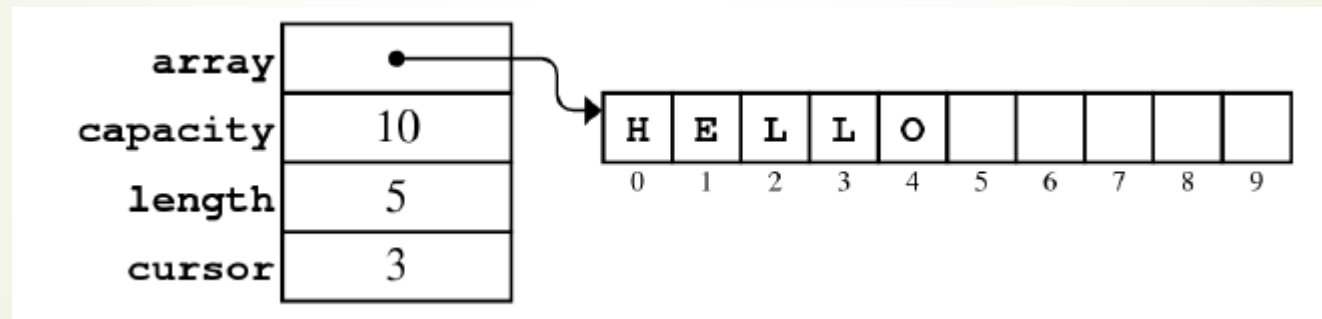
*Ib
  a b c
      ^

*
```

Реализация на буфера с масив

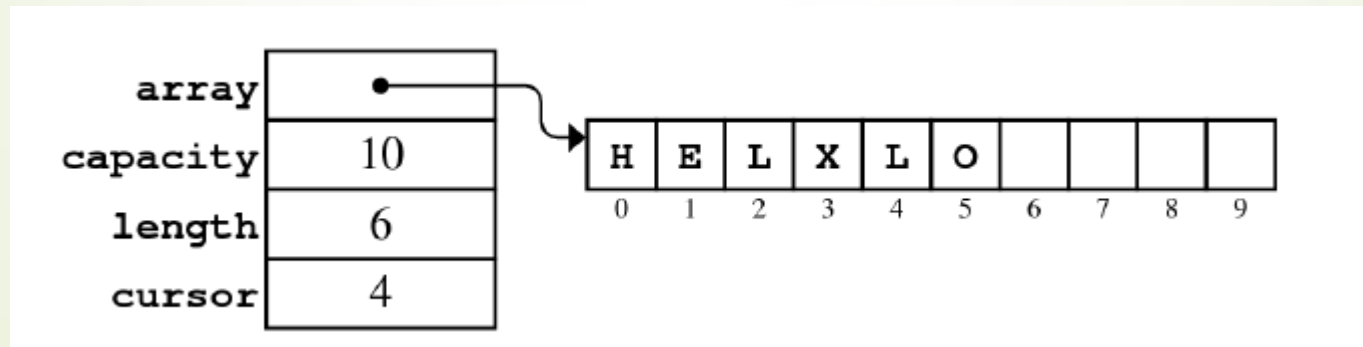
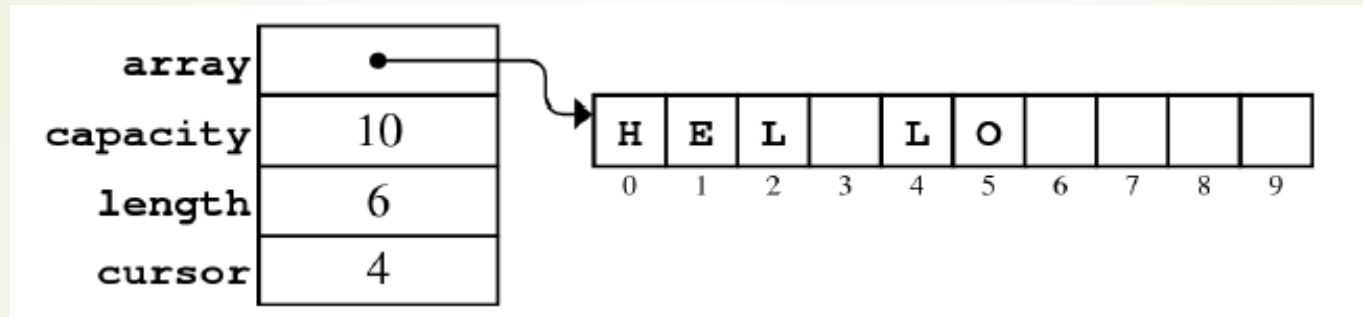
H E L L O
 ^

Паметта изглежда така:



Реализация на буфера с масив

- Как ще вмъкнем **X** след първото **L**?



- А как ще приемем символи?

Дефиниция за O (о-голямо)

За характеризация на сложността на алгоритмите използваме символа O (о-голямо).

Нека са дадените функции $f(N)$ и $g(N)$ дефинирани в множеството на естествените числа и със стойности в това множество.

Казваме, че $f(N)$ е $O(g(N))$, ако съществува $const > 0$ и $N_0 > 0$ такива, че за всяко $N > N_0$ е в сила неравенството $f(N) \leq const * g(N)$.

Много често $g(N)=1$, $g(N)=N$, $g(N)=N^2$, $g(N)=N \log(N)$, а $f(N)$ е броят на операциите за решаването на някаква задача с използване на някакъв алгоритъм.

Тогава казваме, че имаме **изчислителна сложност (computational complexity)** съответно: $O(1)$, $O(N)$, $O(N^2)$, $O(N \log(N))$.

TABLE 13-2 Computational complexity of the array-based buffer

Operation	Array
<code>moveCursorForward</code>	$O(1)$
<code>moveCursorBackward</code>	$O(1)$
<code>moveCursorToStart</code>	$O(1)$
<code>moveCursorToEnd</code>	$O(1)$
<code>insertCharacter</code>	$O(N)$
<code>deleteCharacter</code>	$O(N)$

Реализация на буфера с два стека (before и after)

H E L L O
 ^

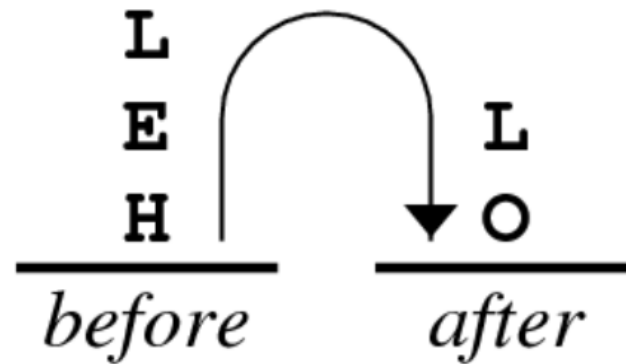


TABLE 13-3 Computational complexity of the array- and stack-based buffers

Operation	Array	Stack
<code>moveCursorForward</code>	$O(1)$	$O(1)$
<code>moveCursorBackward</code>	$O(1)$	$O(1)$
<code>moveCursorToStart</code>	$O(1)$	$O(N)$
<code>moveCursorToEnd</code>	$O(1)$	$O(N)$
<code>insertCharacter</code>	$O(N)$	$O(1)$
<code>deleteCharacter</code>	$O(N)$	$O(1)$

Реализация със свързан списък, в който имаме **dummy node**

$\wedge A B C$

$A \wedge B C$

$A B \wedge C$

$A B C \wedge$

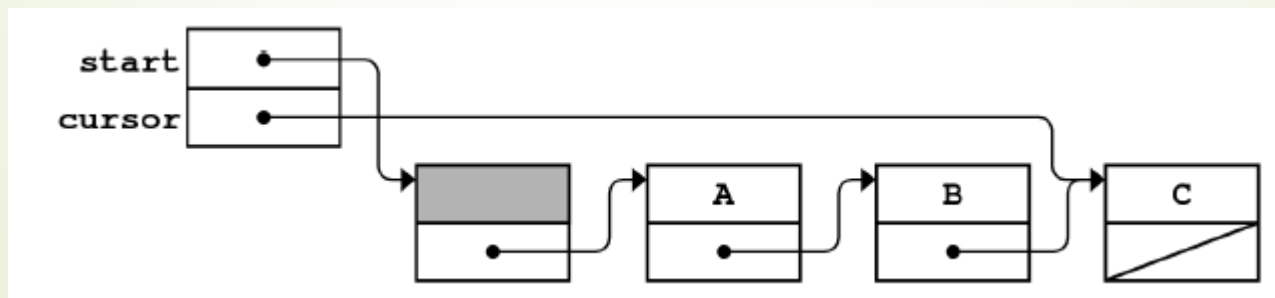
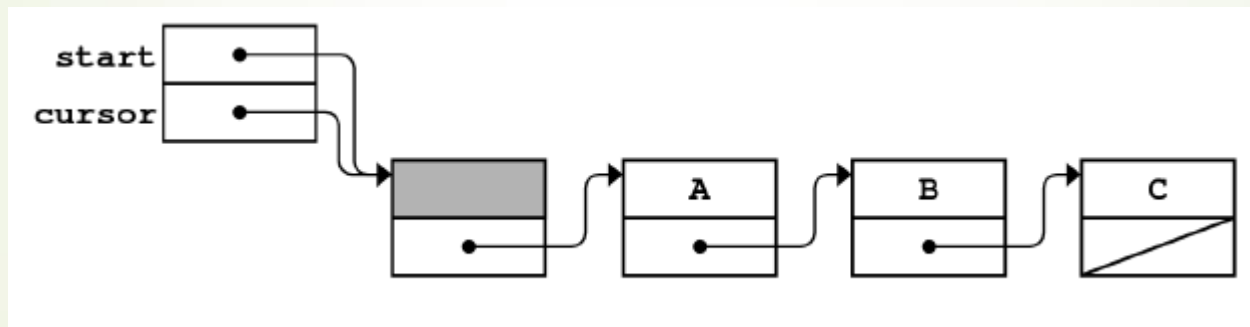
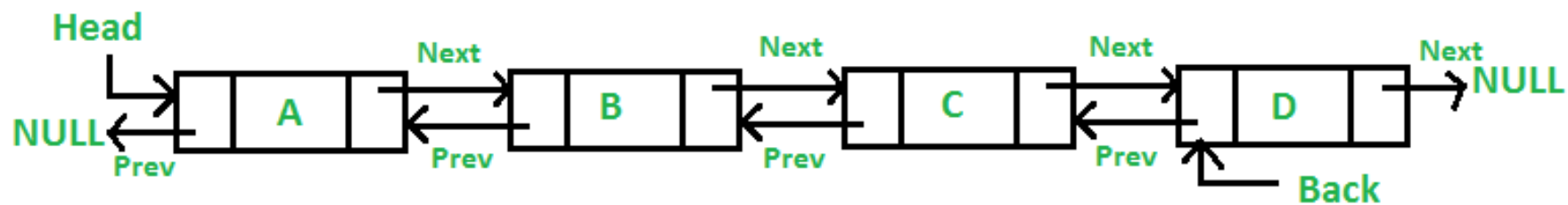


TABLE 13-4 Computational complexity of the three buffer models

Operation	Array	Stack	List
<code>moveCursorForward</code>	$O(1)$	$O(1)$	$O(1)$
<code>moveCursorBackward</code>	$O(1)$	$O(1)$	$O(N)$
<code>moveCursorToStart</code>	$O(1)$	$O(N)$	$O(1)$
<code>moveCursorToEnd</code>	$O(1)$	$O(N)$	$O(N)$
<code>insertCharacter</code>	$O(N)$	$O(1)$	$O(1)$
<code>deleteCharacter</code>	$O(N)$	$O(1)$	$O(1)$

А можем ли да изберем структура данни, така че цялата колона да съдържа $O(1)$?

Отговор: Да. Двусвързан списък (**Doubly Linked List**)



ИЗПОЛЗВАНИ ИЗТОЧНИЦИ

- ➡ http://web.stanford.edu/dept/cs_edu/BXReader-Beta-2012.pdf (chapter 13 “Efficiency and Representation“, страници 570-614)