Lecture No.02
List Implementation via Array

CC-213 Data Structures
Department of Computer Science
University of the Punjab

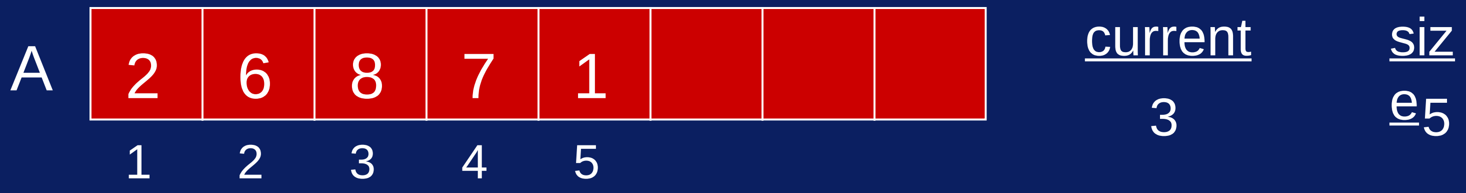
Slides modified very slightly from the late Dr. Sohail Aslam’s lectures at VU

Implementing Lists

- We have designed the interface for the List; we now must consider how to implement that interface.

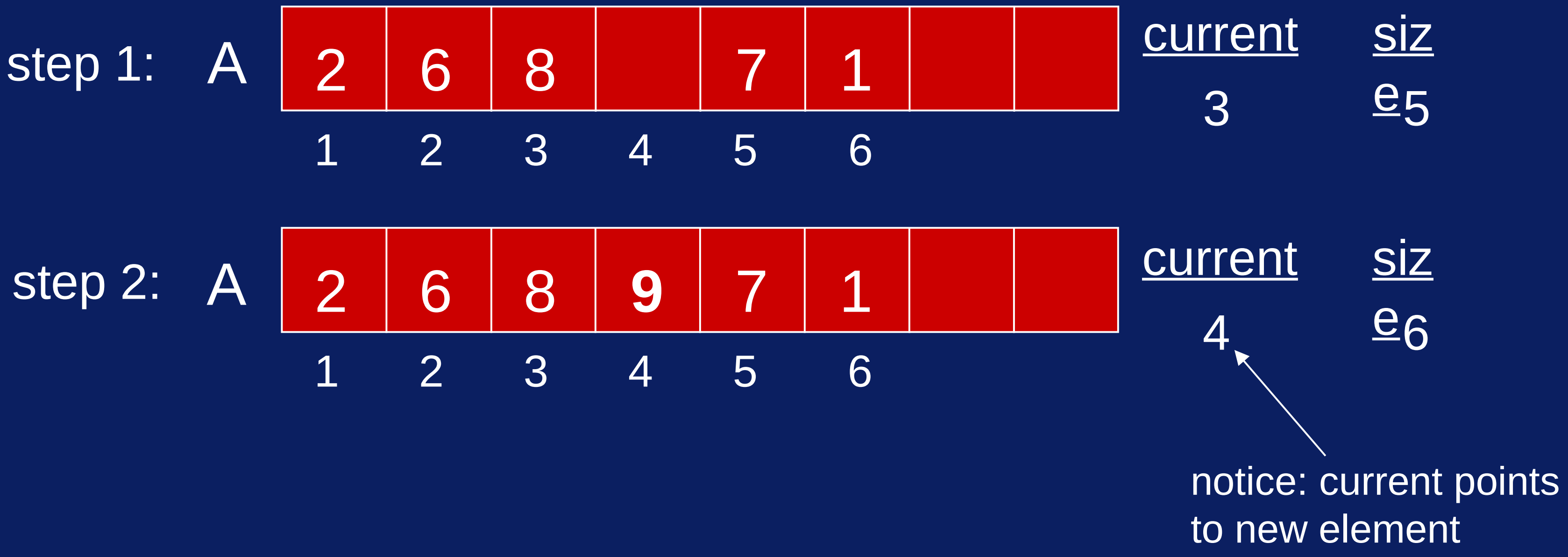
Implementing Lists

- We have designed the interface for the List; we now must consider how to implement that interface.
- Implementing Lists using an array: for example, the list of integers (2, 6, 8, 7, 1) could be represented as:



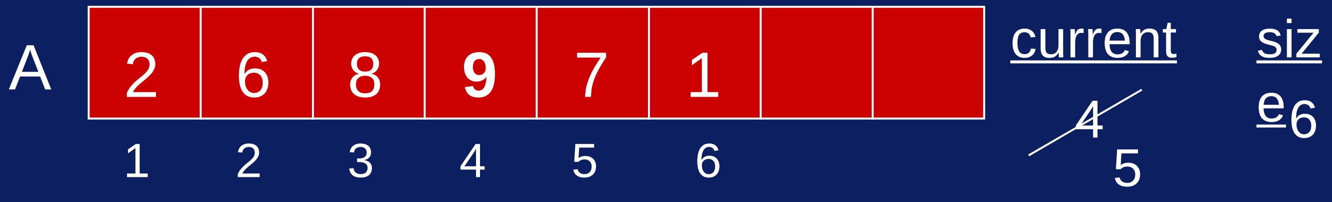
List Implementation

- add(9); current position is 3. The new list would thus be: (2, 6, 8, 9, 7, 1)
- We will need to *shift* everything to the right of 8 one place to the right to make place for the new element '9'.



Implementing Lists

- next():



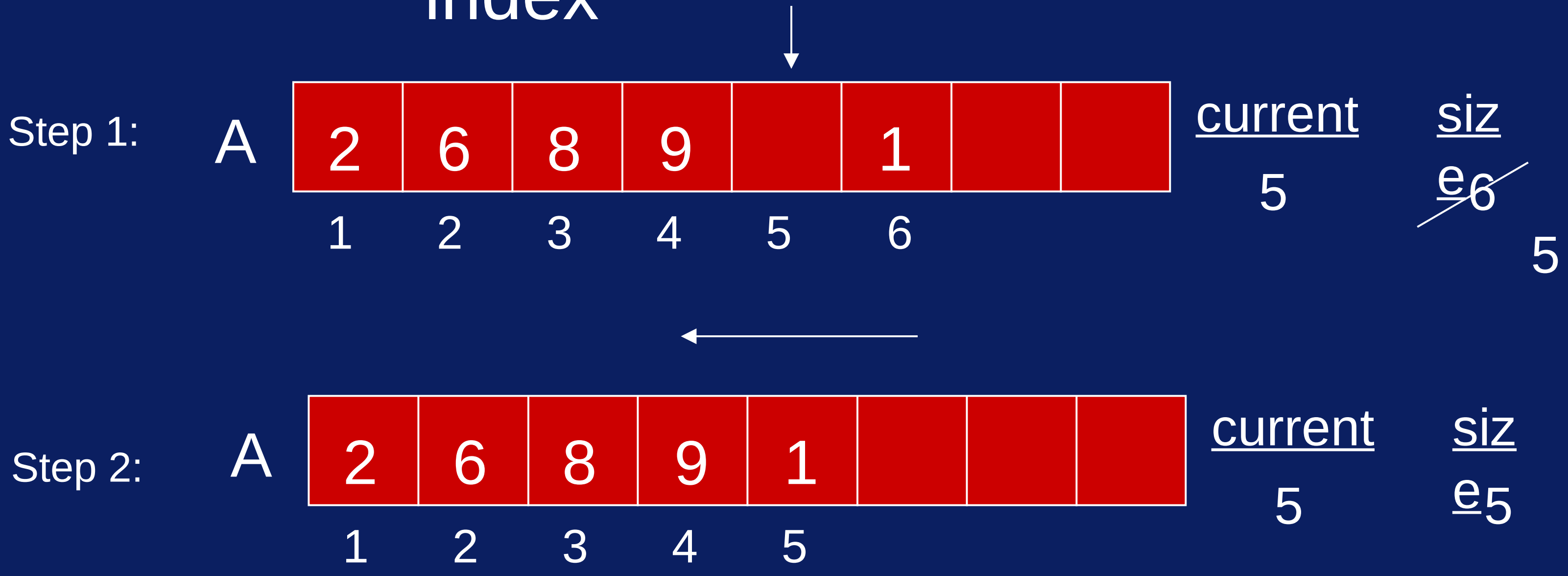
Implementing Lists

- There are special cases for positioning the current pointer:
    a. past the last array cell
    b. before the first cell

- We will have to worry about these when we write the actual code.

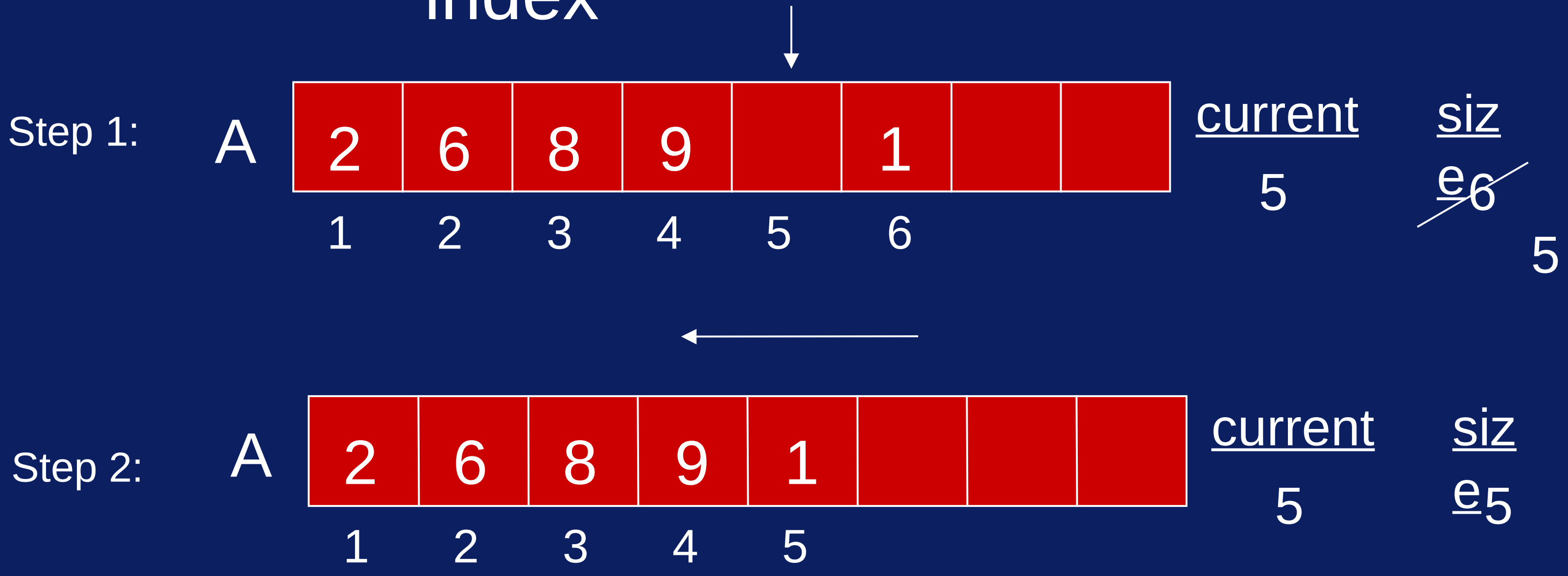
Implementing Lists

- remove(): removes the element at the current index



Implementing Lists

- remove(): removes the element at the current index



- We fill the blank spot left by the removal of 7 by shifting the values to the right of position 5 over to the left one space.

Implementing Lists

find(X): traverse the array until X is located.

```
int find(int X)
{
int j;
for(j=1; j < size+1; j++ )
    if( A[j] == X ) break;

if( j < size+1 ) {  // found X
    current = j;        // current points to where X found
    return 1;   // 1 for true
}
return 0;  // 0 (false) indicates not found
}
```

Implementing Lists

- Other operations:

get()	→ return A[current];
update(X)	→ A[current] = X;
length()	→ return size;
back()	→ current--;
start()	→ current = 1;
end()	→ current = size;

Analysis of Array Lists

- add
  - we have to move every element to the right of current to make space for the new element.
  - Worst-case is when we insert at the beginning; we have to move every element right one place.
  - Average-case: on average we may have to move half of the elements

Analysis of Array Lists

- remove
  - Worst-case: remove at the beginning, must shift all remaining elements to the left.
  - Average-case: expect to move half of the elements.

- find
  - Worst-case: may have to search the entire array
  - Average-case: search at most half the array.

- Other operations are one-step.