Fundamental Building Blocks

Brian G. Patrick

Data Structure

A data structure is an organized collection of data (items),
so done to facilitate
their storage, retrieval, and manipulation.

Every data structure we see in this course is constructed using only two types of building blocks

Which two?

Array and Node

Array

Array

Definition

- An array is a linear collection of homogeneous objects (items).

What is a linear collection?

- Objects are stored one after each other in memory.

What is homogeneous?

- All objects have the same type.

In C#

- An array is a reference type.
- Array indices are zero-based, i.e. indices start at 0.

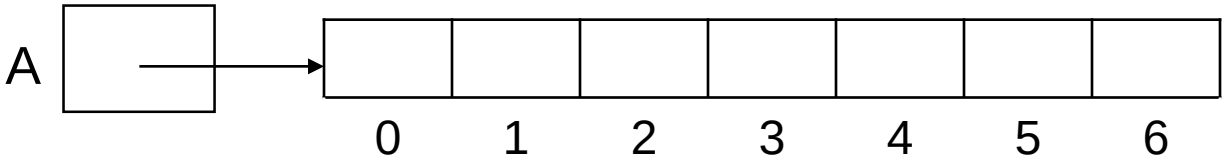
Example

Declaration of A as (a reference to) an array of integers

int [ ] A;

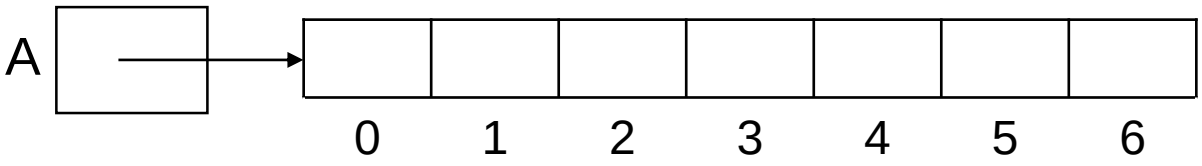
Definition of A as (a reference to) an array object that can hold 7 integers

A = new int [7];



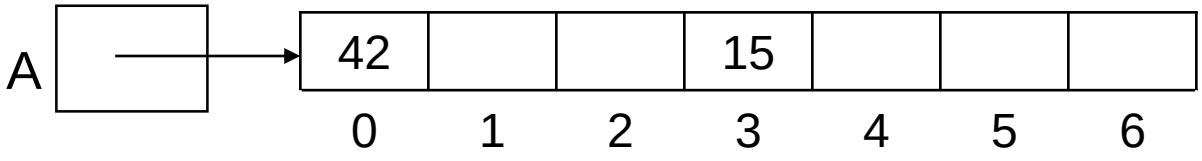
Declaration and definition of A in one step

int [ ] A = new int [7];



Accessing A (using indices)

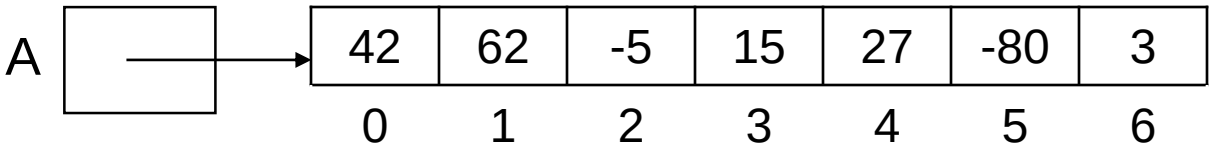
A[0] = 42;
A[3] = 15;
int x = A[3];



Useful property

A.Length which returns the capacity of array A, in this case, 7.

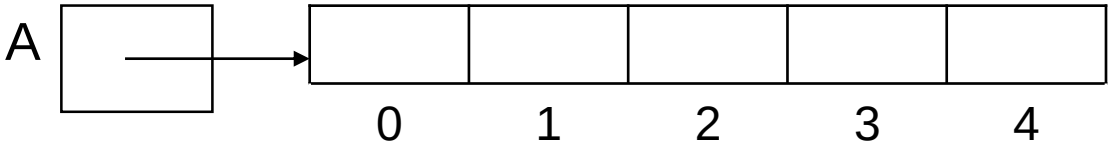
Adding up the integers in A




```
int sum = 0;
for (int i = 0; i < A.Length; i++)
    sum += A[i];

Console.Write(sum);        // 64
                           // Everything is base 2 in Computer Science ☺
```

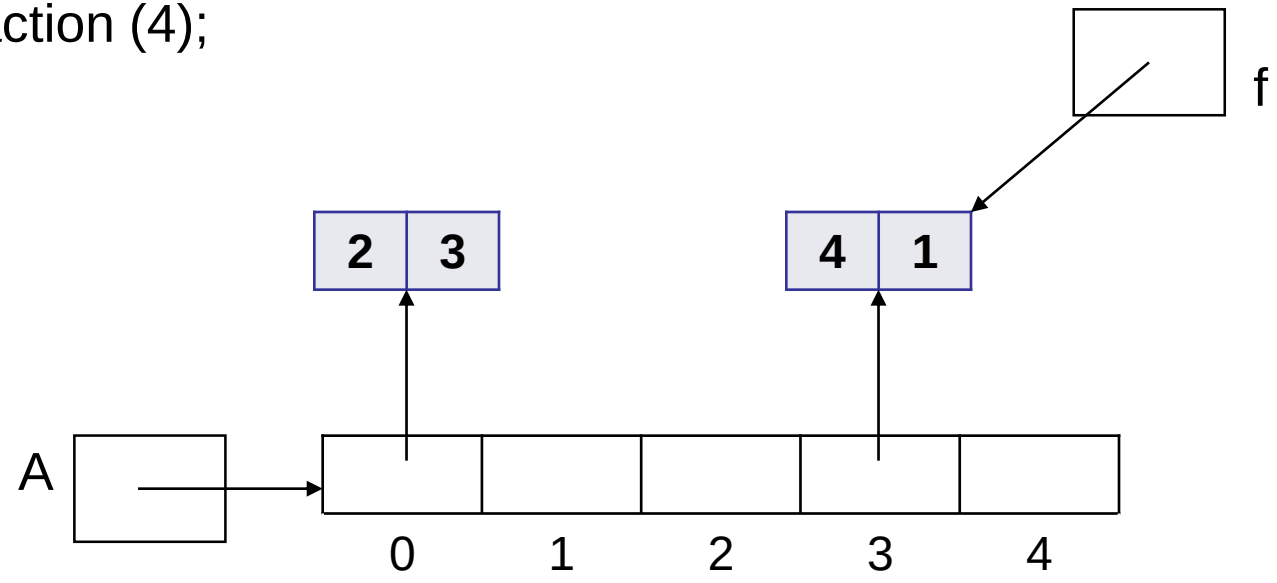
Example

```
Fraction [ ] A;                    // Declaration of array A
A = new Fraction [5];              // Definition of array A

or

int [ ] A = new Fraction [5];      // Declaration + definition of A
```



How much space is set aside for each location in A?

A[0] = new Fraction (2,3);
A[3] = new Fraction (4);




```
int x = A[0].Num;        // using the Num property of Fraction
int y = A[3].Den;        // using the Den property of Fraction
Fraction f = A[3];       // f and A[3] now refer to the same fraction
```

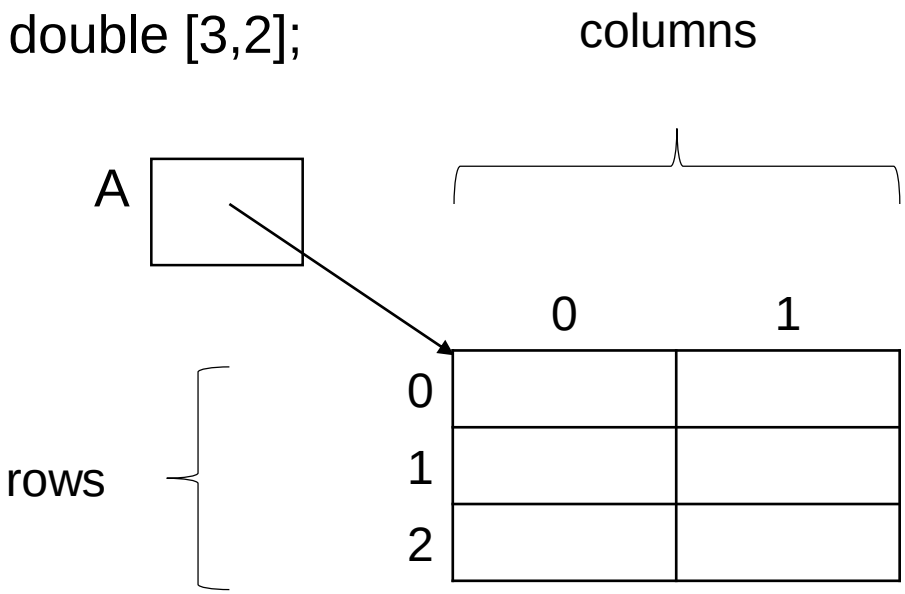
Two-Dimensional Arrays

Rectangular Array

Each row has the same number of columns.

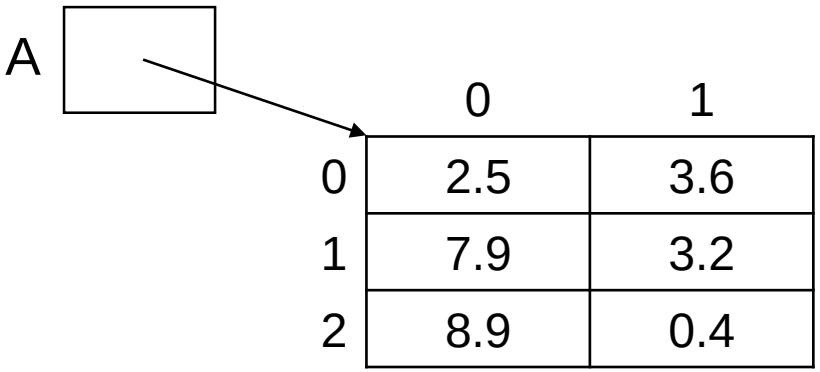
Declare + define

```
double [ , ] A;
A = new double [3,2];
```



Declare + define + initialize

double [,] A = { {2.5, 3.6}, {7.9, 3.2}, {8.9, 0.4} };



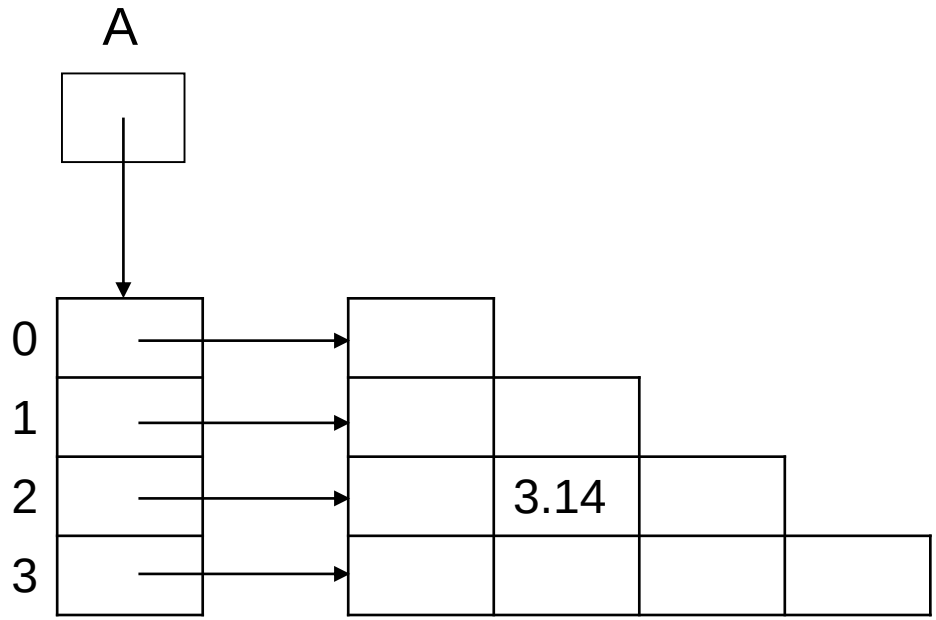
Accessing A (using two indices)

double x = A[2,0]; // 8.9

Jagged Array

Each row may have a different number of columns.

```
float [ ][ ] A;
A = new float [4][ ];
for (int i = 0; i < 4; i++)
    A[i] = new float [i+1];

A[2][1] = 3.14;
```



Things to bear in mind:

- An incorrect index will generate an IndexOutOfRangeException.
- The capacity of the linear array is fixed when it is declared.
- To increase the capacity of an array A requires a bit of work.
  - A new array is defined and declared with greater capacity,
  - The items of A are copied over to the new array, and
  - A is assigned to the new array.

Node

Node

Definition

- A node is a class (reference type) that encapsulates:
  - one or more items which are not necessarily of the same type.
  - one or more references to other nodes.

- A node is used as part of a data structure such as a linked list or a binary tree that can grow and shrink as needed during the execution of a program.

Generic Class Node

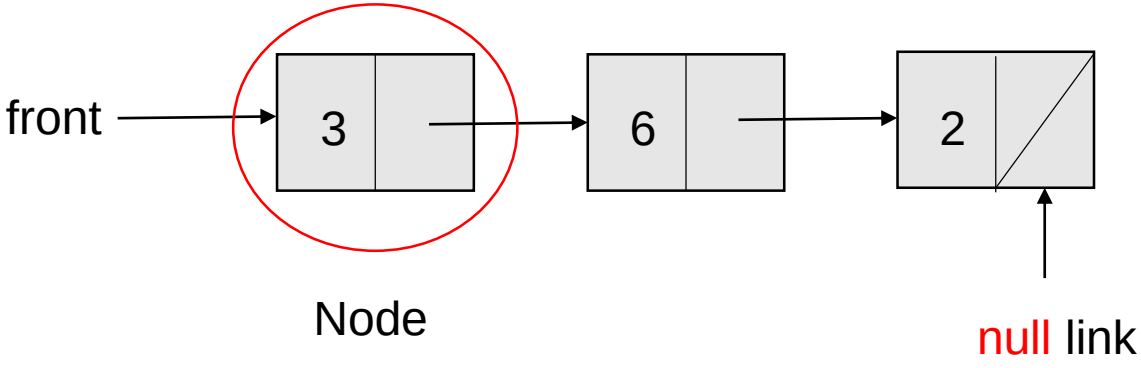
```
class Node<T>
{
    private T item;
    private Node<T> next;

    public Node(T item, Node<T> next = null)
    {    this.item = item; this.next = next;  }
    ...
}
```

- The Node class will often appear (in this course) within another class. In this case, the Node class is private, but its data members are public.

```
class C<T>
{
        private Node
        {
            public T item;
            public Node next;
            ...
        }

        private Node front;
        ...
}
```

Example

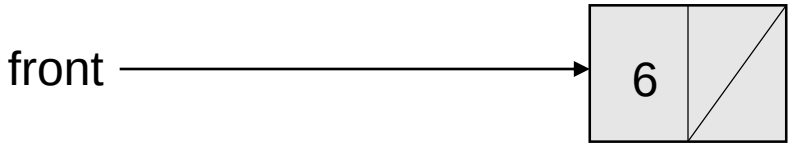
- Look at the following diagram where each of the three nodes contains an integer and a “link” (reference) to the next node.

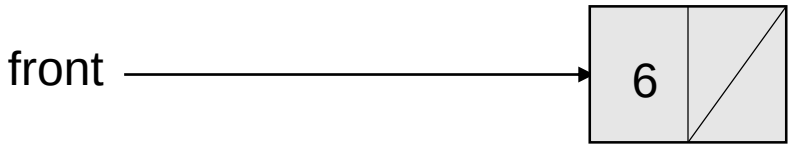


Let’s build it!

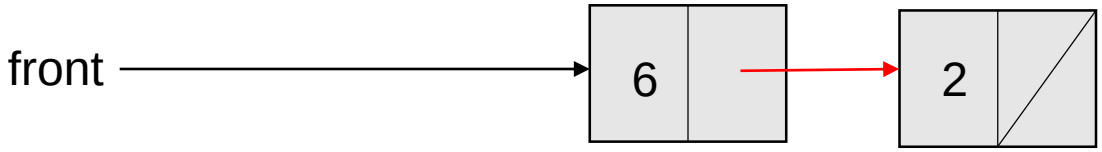
```
Node<T> front = null;          // front is declared and initialized to null
```

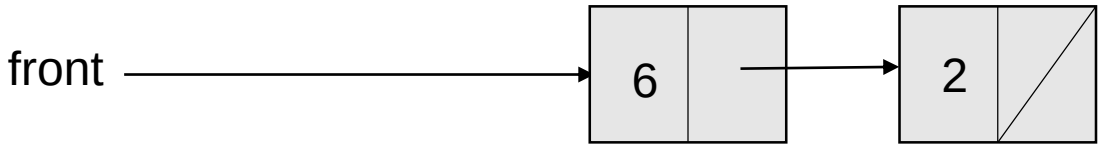

```
front = new Node<int>(6);     // front now refers to a node containing 6
```



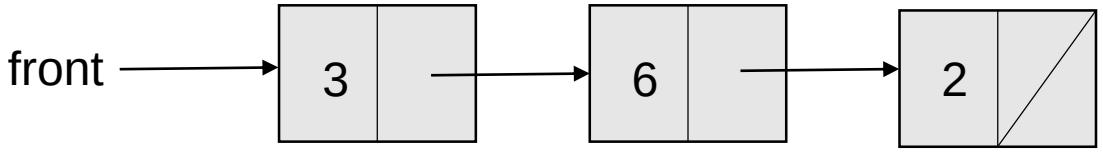


front.next = new Node<int>(2);





front = new Node<int>(3,front);



On to building ...