Lecture 26

- Covers
  - Array basics
- Reading: Savitch 6.1

Example

- Write a program to read in 5 integers from the user and display a horizontal bar chart for the values
- E.g.

```
Enter 5 integers: 5 6 2 4 3
* * * * *
* * * * * *
* *
* * * *
* * *
```

Example

```
public static void main(String[ ] args)
{
   Scanner keyboard = new Scanner(System.in);
    System.out.print("Enter 5 integers: ");
    int num1 = keyboard.nextInt( );
    int num2 = keyboard.nextInt( );
    int num3 = keyboard.nextInt( );
    int num4 = keyboard.nextInt( );
    int num5 = keyboard.nextInt( );

    for (int i = 0; i < num1; ++i)
    {
        System.out.print("* ");
    }
    System.out.println();

    for (int i = 0; i < num2; ++i)
    {
        System.out.print("* ");
    }
    System.out.println( );
```

Example

```
    for (int i = 0; i < num3; ++i)
    {
        System.out.print("* ");
    }
    System.out.println( );

     for (int i = 0; i < num4; ++i)
    {
        System.out.print("* ");
    }
    System.out.println( );

    for (int i = 0; i < num5; ++i)
    {
        System.out.print("* ");
    }
    System.out.println( );
}
```

Example

- We can rewrite this bar chart program because we do not need to keep each input value after we have output the corresponding bar chart

Example

```
public static void main(String[ ] args)
{
    System.out.print("Enter 5 integers: ");
    Scanner keyboard = new Scanner(System.in);
    int num;
    for (int bar = 0; bar < 5; ++bar)
    {
        num = keyboard.nextInt( );
        for (int i = 0; i < num; ++i)
        {
            System.out.print("* ");
        }
        System.out.println( );
    }
}
```

Example 2

- Write a program to read in 5 integers from the user and display a vertical bar chart for the values
- E.g.

```
Enter 5 integers: 3 5 4 6 9
          *
          *
          *
        * *
    *   * *
    * * * *
  * * * * *
  * * * * *
  * * * * *
```

Example 2

- Algorithm

Get the 5 integers
Find the maximum of the 5 integers
LOOP FOR i from the maximum to greater than 0
IF first integer is >= i THEN
Output " "*
ELSE
Output 2 spaces
ENDIF
IF second integer is >= i THEN
Output " "*
ELSE
Output 2 spaces
ENDIF
etc. for third, fourth and fifth integers
Output newline
ENDLOOP

Example 2

```
public static void main(String[ ] args)
{
    Scanner keyboard = new Scanner(System.in);
    System.out.print("Enter 5 integers: ");
    int num1 = keyboard.nextInt( );
    int num2 = keyboard.nextInt( );
    int num3 = keyboard.nextInt( );
    int num4 = keyboard.nextInt( );
    int num5 = keyboard.nextInt( );

    max = maximum(num1,num2);
    max = maximum(max,num3);
    max = maximum(max,num4);
    max = maximum(max, num5);
```

Example 2

```
for (int i = max; i > 0; --i)
{
    if (num1 >= i)
    {
        System.out.print("* ");
    }
    else
    {
        System.out.print("  ");
    }
    if (num2 >= i)
    {
        System.out.print("* ");
    }
    else
    {
        System.out.print("  ");
    }
```

Example 2

```
if (num3 >= i)
{
    System.out.print("* ");
}
else
{
    System.out.print("  ");
}
if (num4 >= i)
{
    System.out.print("* ");
}
else
{
    System.out.print("  ");
}
```

Example 2

```
        if (num5 >= i)
        {
            System.out.print("* ");
        }
        else
        {
            System.out.print("  ");
        }
        System.out.println( );
    }
```

Example 2

- We cannot rewrite the vertical bar chart in the same way as the horizontal bar chart as we need to keep track of all the numbers throughout
- What happens if we want to display a vertical bar chart for 50 numbers?
- We would need to store all 50 values

Arrays

- When we need to store a collection of related values like this, it is inconvenient to name them all individually
- Arrays allow us to deal with a collection of related values (of the same type) together
- We give the collection a name, and refer to each variable in the collection by a number called its index

Arrays

- Arrays – a way of declaring and using a collection of related variables
- Creating arrays

```
int[ ] intArray = new int[5];
double[ ] doubleArray = new double[100];
String[ ] stringArray = new String[8];
```

Arrays

- The declaration

  ```
  int[ ] intArray = new int[5];
  ```

  allocates memory for a collection (array) of integers
- The collection of integer values is referred to through the array name intArray
- Each element (value) in the array is specified by giving the name of the array and its position in the array (index)

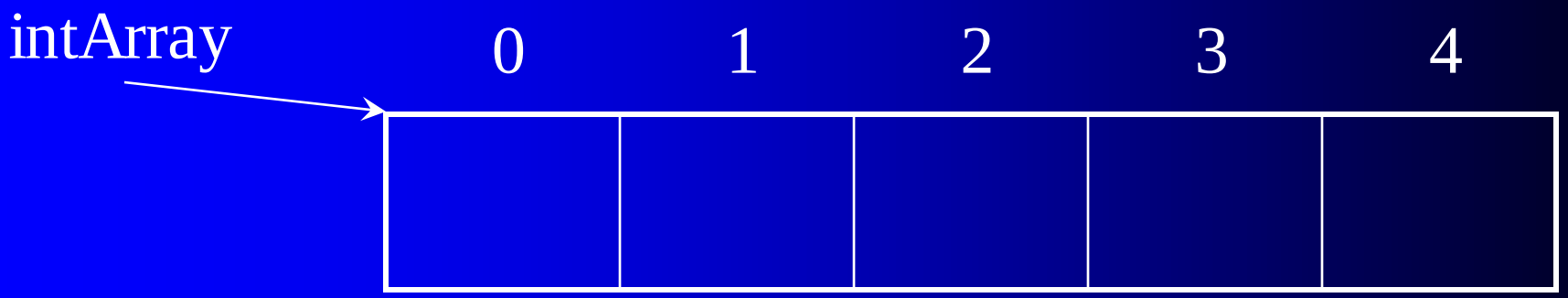
Arrays

```
int[ ] intArray = new int[5];
```

- Arrays are created almost like an object but
  - The name of the base type follows the keyword new
  - The number of elements in the array is specified in square brackets

Array indexes

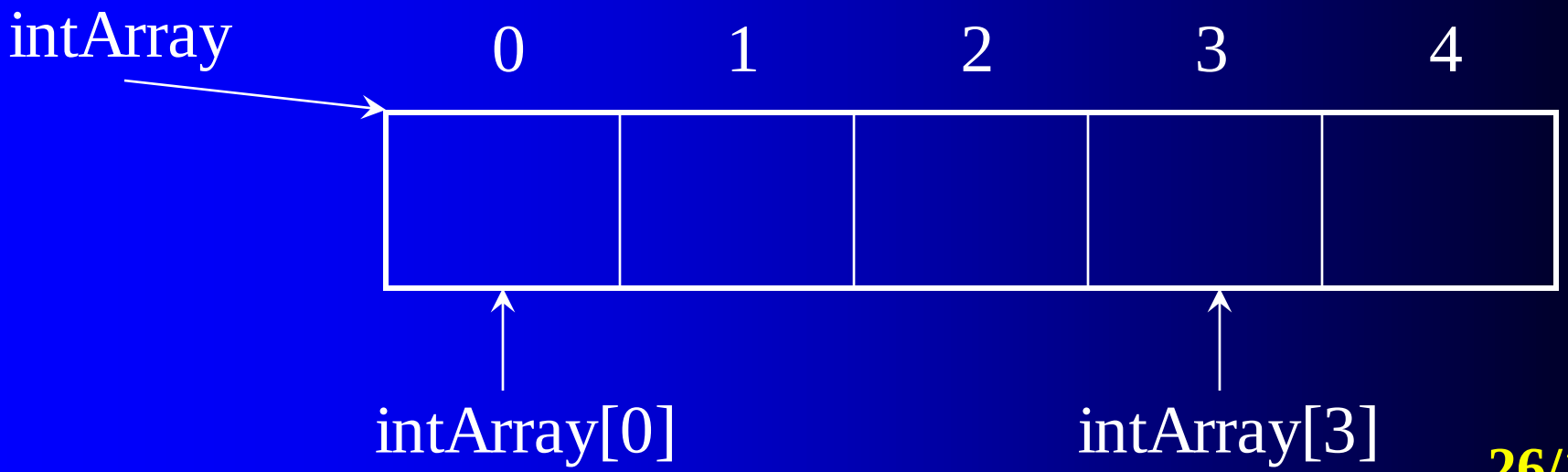
```
int[ ] intArray = new int[5];
```



- Indexes number each element in the array
- They always start from 0

Subscripting operators

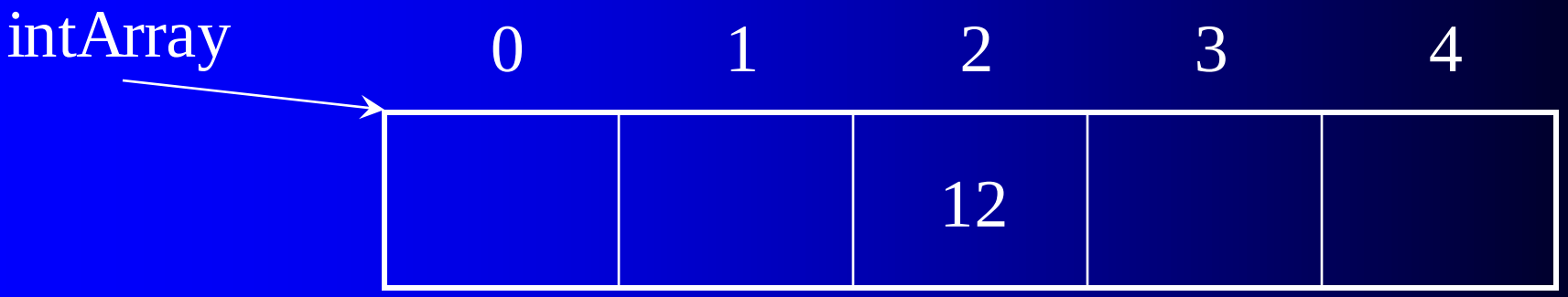
- To refer to an individual element in the array, we use the name of the array and the index number specified with the subscripting operator



Assigning values to array elements

- To assign a value to an array element, we specify the array element on the lhs of an assignment operator

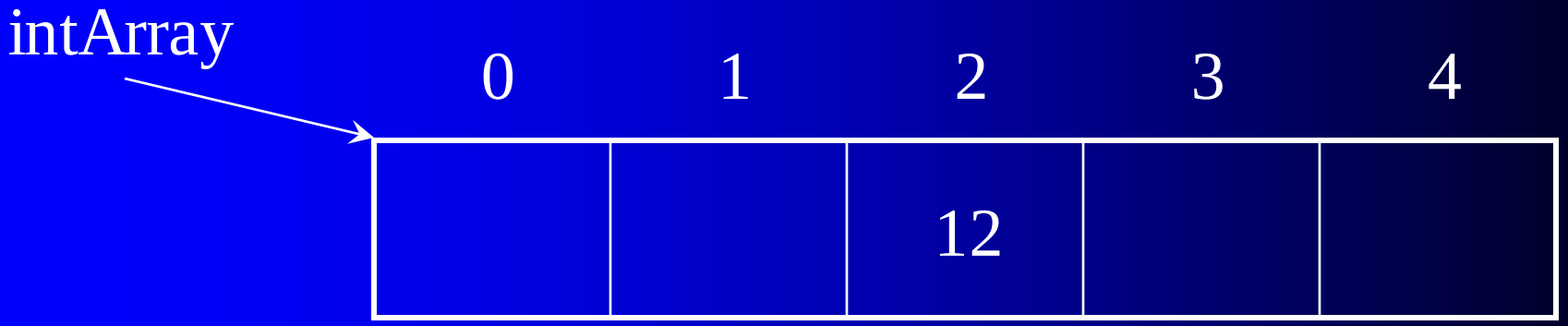
```
intArray[2] = 12;
```



Retrieving values from array elements

- To use the value stored in an array element, we refer to the variable through with the subscripting operators

```
int num = intArray[2];
System.out.println(intArray[2]);
```



Example (revisited)

- Change the horizontal bar chart program so that it uses an array
- E.g.

```
Enter 5 integers: 5 6 2 4 3
* * * * *
* * * * * *
* *
* * * *
* * *
```

Example (revisited)

```
public static void main(String[ ] args)
{
    Scanner keyboard = new Scanner(System.in);
    System.out.print("Enter 5 integers: ");

    int [ ] number = new int[5];
    for (int i = 0; i < 5; ++i)
    {
        number[i] = keyboard.nextInt( );
    }

    for (int i = 0; i < 5; ++i)
    {
        for (int j = 0; j < number[i]; ++j)
        {
            System.out.print("* ");
        }
        System.out.println( );
    }
}
```

Example 2 (revisited)

- Change the vertical bar chart program so that it uses an array
- E.g.

```
Enter 5 integers: 3 5 4 6 9
          *
          *
          *
        * *
  *     * *
  * * * * *
* * * * * *
* * * * * *
* * * * * *
```

Example 2 (revisited)

```
public static void main(String[ ] args)
{
    Scanner keyboard = new Scanner(System.in);
    System.out.print("Enter 5 integers: ");
    int [ ] number = new int[5];
    for (int i = 0; i < 5; ++i)
    {
        number[i] = keyboard.nextInt( );
    }
    int max = maximum(maximum(number[0], number[1]),
                      maximum(number[2],
                              maximum(number[3],number[4])));
```

Example 2 (revisited)

```
for (int i = max; i > 0; --i)
{
    for (int j = 0; j < 5; ++j)
    {
        if (number[j] >= i)
        {
            System.out.print("* ");
        }
        else
        {
            System.out.print("  ");
        }
    }
    System.out.println( );
}
}
```

Creating arrays

- When creating an array, the size of the array can be specified with an integer literal, an integer variable, or an expression that evaluates to an integer

```
int size = 10;
double[ ] doubleArray = new double[size];
boolean[ ] booleanArray = new boolean[10];
int[ ] intArray = new int[size * 2 + 1];
```

Creating arrays

- The size of the array is stored as a (public) attribute of the array and can be accessed with the name of the array, the dot operator and the attribute name length

  doubleArray.length

- The size of an array object, once created, cannot be changed

Example 2 (revisited yet again)

- Rewrite the vertical bar chart program so that it reads the number of values to be entered from the user and creates an array of the appropriate size
- E.g.

```
Enter the number of values in the bar chart: 7
Enter 7 integers: 2 3 4 5 2 1 7
                  *
                  *
            *     *
         * *      *
      * * *       *
   * * * * *      *
* * * * * * *     *
```

Example 2 (revisited yet again)

```
public static void main(String[ ] args)
 {
    System.out.print("Enter the number of values in the bar chart: ");
    int arraySize = keyboard.nextInt( );


    int[ ] number = new int[arraySize];

    System.out.print("Enter " + arraySize + " integers: ");
    for (int i = 0; i < arraySize; ++i)
    {
       number[i] = keyboard.nextInt( );
    }

    int max = number[0];
    for (int i = 1; i < number.length; ++i)
    {
       if (number[i] > max)
          max = number[i];
    }
```

Example 2 (revisited yet again)

```
    for (int i = max; i > 0; --i)
    {
        for (int j = 0; j < number.length; ++j)
       {
            if (number[j] >= i)
            {
             System.out.print("* ");
            }
            else
            {
             System.out.print("  ");
            }
        }
        System.out.println( );
    }
}
```

Array index out of bounds

- If we attempt to access an array outside its bounds, we get a run-time "Array index out of bounds" error
- E.g.

```
int[ ] num = new int[10];
for (int j = 0; j <= num.length; ++j)
{
      num[j] = keyboard.nextInt( );
}
```

** On the last iteration through the loop, we try to assign a value to num[10], whereas the indexes must be from 0 to 9*

Initialising arrays

- If we know the initial values of the elements in an array, we can initialise the array

```
int[ ] nums = {12, 10, 24, 31};
```

- The size of the array is not specified here, instead the compiler creates the smallest possible array that will fit the numbers in the initialiser list

```
System.out.println(nums.length);
```

Next lecture

- Programming with arrays