Java: Methods & Strings

B.Sc. 2nd Semester

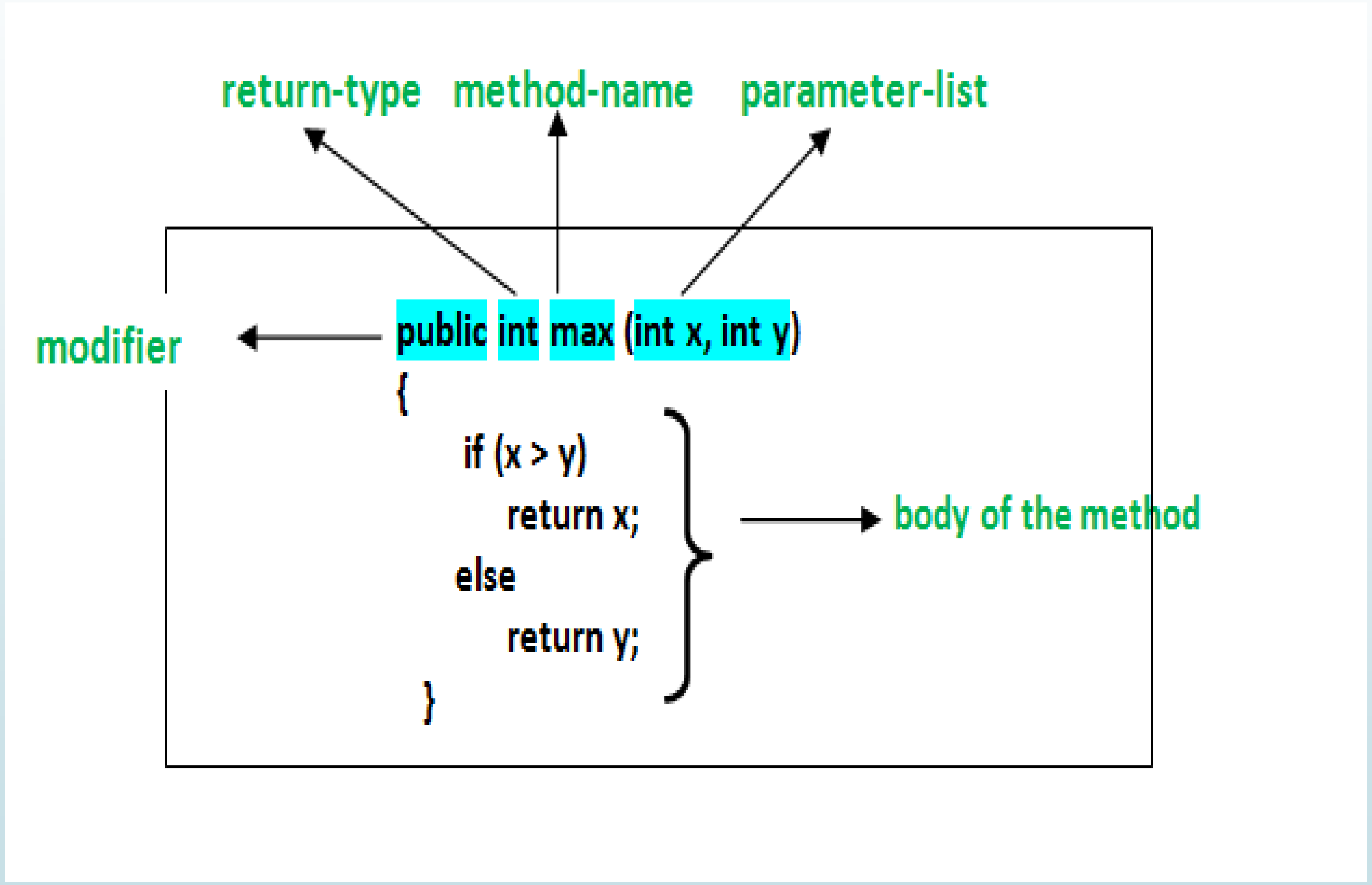
Department of Computer Science and Applications (Paper Code:CC3)

Prof. Paulami Basu Ray

Contents

- Methods
- String Class
- Command Line Arguments
- Common Methods of the String Class

Method



```
public int max (int x, int y)
{
    if (x > y)
        return x;
    else
        return y;
}
```

Method

- A method is a collection of statements that perform some specific task and return the result to the caller. A method can perform some specific task without returning anything. Methods allow us to **reuse** the code without retyping the code. In Java, every method must be part of some class which is different from languages like C, C++.
  Methods are **time savers** and help us to **reuse** the code without retyping the code.
- **Method Declaration**

In general, method declarations has six components :

Method

1. Modifier-: Defines **access type** of the method i.e. from where it can be accessed in your application. In Java, there 4 type of the access specifiers.:

- public: accessible in all class in your application.
- protected: accessible within the class in which it is defined and in its **subclass(es)**
- private: accessible only within the class in which it is defined.
- default (declared/defined without using any modifier) : accessible within same class and
- package within which its class is defined.

Method

2. The return type : The data type of the value returned by the method or void if does not return a value.

3. Method Name : The rules for field names apply to method names as well, but the conventi is a little different.

4. Parameter list : Comma separated list of the input parameters are defined, preceded with their data type, within the enclosed parenthesis. If there are no parameters, you must use empty parentheses ().

5. Exception list : The exceptions you expect by the method can throw, you can specify these exception(s). [This is optional]

6. Method body : It is enclosed between braces. The code you need to be executed to perfor your intended operations.

Strings

- The String class represents character strings. All string literals in Java programs, such as "abc", are implemented as instances of this class.
- Strings are constant; their values cannot be changed after they are created.
- String buffers support mutable strings. Because String objects are immutable they can be shared.

- For example:

String str = "abc";

is equivalent to:

char data[] = {'a', 'b', 'c'};
String str = new String(data);

- Here are some more examples of how strings can be used:

```
System.out.println("abc");
String cde = "cde";
System.out.println("abc" + cde);
```

String Conversions

- Try to compile the following code:

```
  class StrConvert{
  public static void main(String args[]){
  String strTest=“100”;
  System.out.println(“Using String:”+(strTest/4));
  }
}
```

Error!

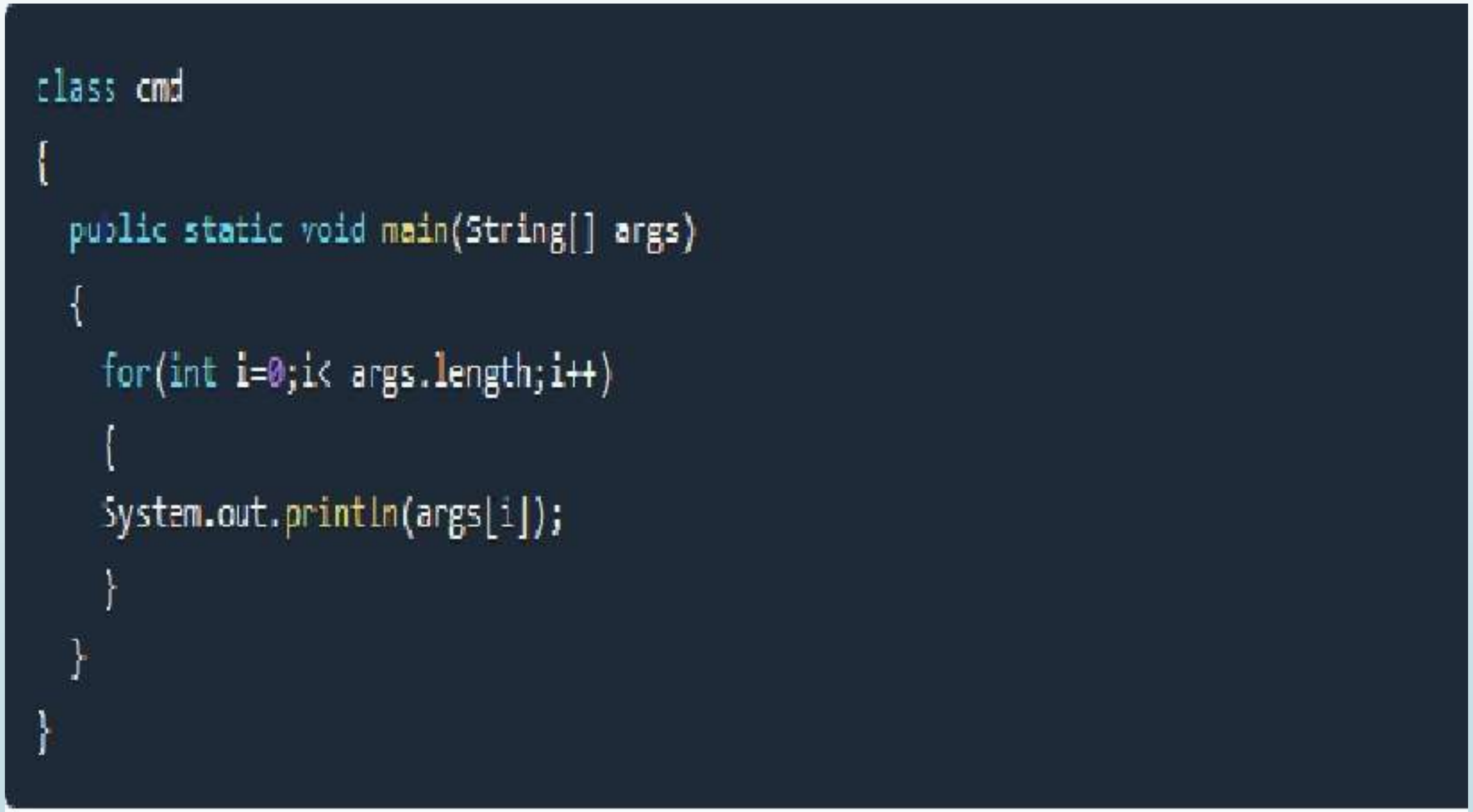
Bad operand types for binary operator

String Conversions

```
class StrConvert
{
        public static void main(String []args)
        {
        String strTest = "100";
        int  IntTest = Integer.parseInt(strTest);
        System.out.println("Actual String:"+ strTest);
        System.out.println("Converted to Int:" + iTest); //This will now show arithmetic op
        System.out.println("Arithmetic Operation on Int: " + (iTest/4));
         }
}
```

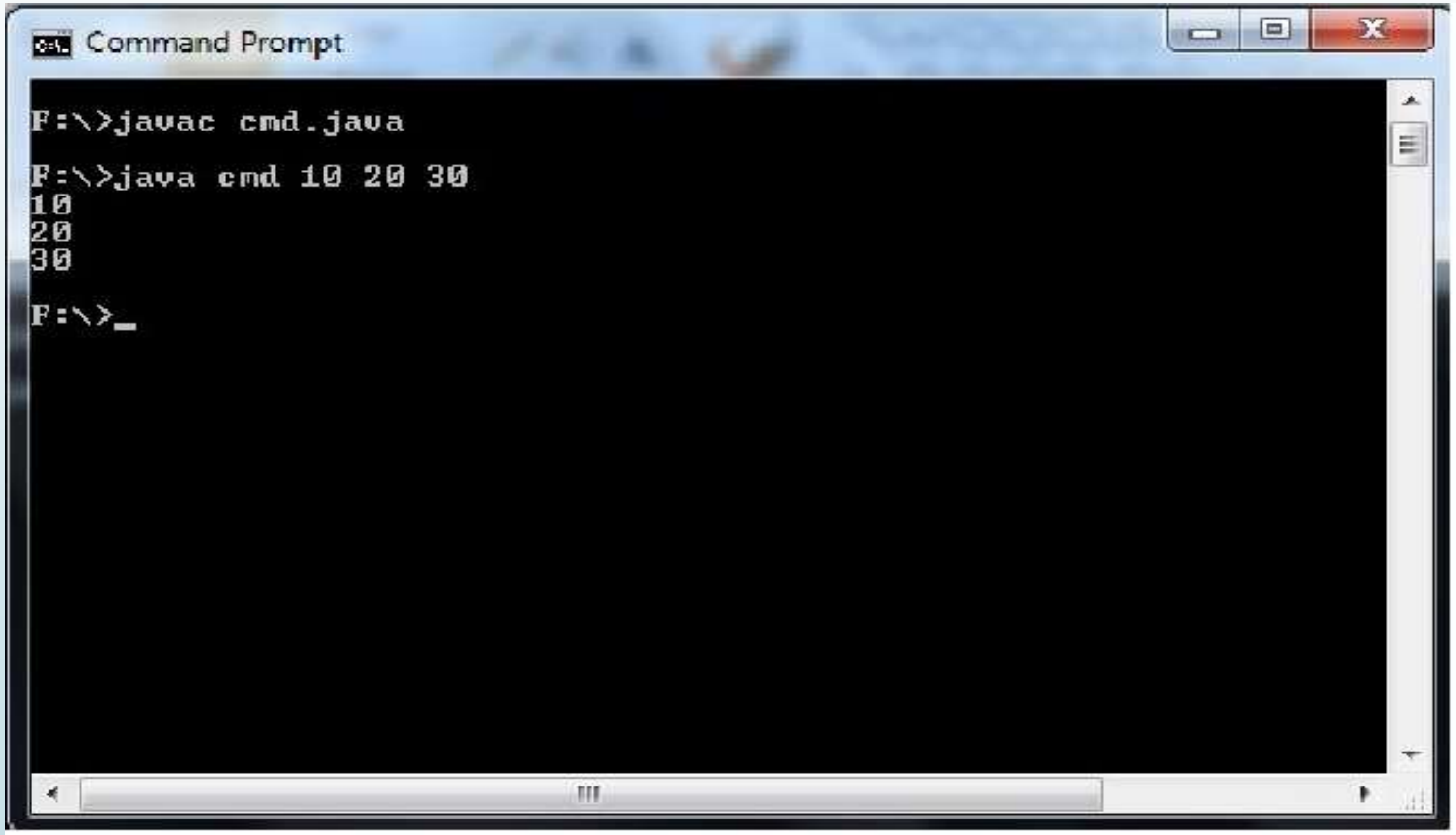
Command Line Arguments

- A **command-line argument** is an information that directly follows the program's name on the **command line** when it is executed.



```
class cmd
{
  public static void main(String[] args)
  {
    for(int i=0;i< args.length;i++)
    {
    System.out.println(args[i]);
    }
  }
}
```

Command Line Arguments



```
F:\>javac cmd.java

F:\>java cmd 10 20 30
10
20
30

F:\>_
```

String Class Methods

Method	Description
char charAt(int index)	returns char value for the particular index
int length()	returns string length
String substring(int beginIndex)	returns substring for given begin index.
String substring(int beginIndex, int endIndex)	returns substring for given begin index and end index.
Boolean contains(CharSequence s)	returns true or false after matching the sequence of char value.
boolean equals(Object another)	checks the equality of string with the given object.
boolean isEmpty()	checks if string is empty.
String concat(String str)	concatenates the specified string.
static String equalsIgnoreCase(String another)	compares another string. It doesn't check case.