Lecture 13: Java Exceptions and dealing with Files.

By Ahmed E. Khaled
Recap to the first part!

- What we discussed so far?
- What are the topics of the second part?
Lecture 13 : Exceptions and files

- Java Exceptions.
- Dealing with files.
Java Exceptions

• An exception is a *problem* that arises during the execution of a program.

• When an Exception occurs the normal flow of the program is *disrupted* and your program terminates *abnormally*.

• To avoid such behavior, these exceptions must be *handled*. 
public static void main(String[] args) {

    int[] myarray = {1,2,3,4};

    Scanner input = new Scanner(System.in);

    int k = input.nextInt();

    System.out.println("myarray at index " + k + " holds "+myarray[k]);
}
• An exception can occur for many different reasons, e.g.:

1. Invalid user input.
2. Your program cannot find the file pointed by user to open.
3. Losing network connection in the middle process.
4. Your program used to much memory.
5. Wrong calculation from your program.
If the user inserted 5, the compiler will throw exception (problem) and terminate your program.

run:
5

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5
  at cop2800.mainclass.main(mainclass.java:27)

Java Result: 1
BUILD SUCCESSFUL (total time: 1 second)
How to catch an exception!

- Your program catches an exception with try and catch.

- A try-catch pair is placed around the code that might generate an exception making it protected from exceptions.

```java
try {
    // Protected code
    // Catch block
}
```
• When an exception occurs, that exception occurred is handled by *catch block* associated with it.

• Every try block should be immediately followed by one or more catch blocks where each declares the type of exception to catch.

• If an exception occurs in protected code, the catch block (or blocks) that follows the try is checked. If the type of exception that occurred is listed in a catch block, the exception is passed to the catch block much as an argument is passed into a method parameter.
try {
    // statements that may cause an exception
}
catch (exception(type)1 e(object)) {
    // error handling code
}
catch (exception(type)2 e(object)) {
    // error handling code
}
catch (exception(type)3 e(object)) {
    // error handling code
}
Take an example

```c
int num1=0;
int num2=0;
num2 = 62 / num1;
```
Exception in thread "main" java.lang.ArithmeticException: / by zero
  at cop2800.mainclass.main(mainclass.java:25)
Java Result: 1
BUILD SUCCESSFUL (total time: 0 seconds)
int num1=0;
int num2=0;
try {
    // Try block to handle code that may cause exception

    num2 = 62 / num1;

    System.out.println("Try block message");
}

} catch (ArithmeticException e) {
    // This block is to catch divide-by-zero error
    System.out.println("Error: Don't divide a number by zero");
}

System.out.println("I'm out of try–catch block in Java.");
run:
Error: Don't divide a number by zero
I'm out of try-catch block in Java.
BUILD SUCCESSFUL (total time: 3 seconds)
try{

    int[] myarray = {1,2,3,4};

    Scanner input=new Scanner(System.in);

    int k = input.nextInt();

    System.out.println("myarray at index " + k + " holds "+myarray[k]);

} catch(ArrayIndexOutOfBoundsException aob){

    System.out.println("Something went wrong with your input, out of bound !");

}
run:
5
Something went wrong with your input, out of bound!
BUILD SUCCESSFUL (total time: 7 seconds)
Java allows to declare general catch for all types of problems.

```java
public static void main(String[] args) {
    try{
        int[] myarray = {1,2,3,4};
        Scanner input = new Scanner(System.in);
        int k = input.nextInt();
        System.out.println("myarray at index " + k + " holds " + myarray[k]);
    }
    catch(Exception exc){
        System.out.println("Something went wrong with your input!");
        System.out.println(exc);
    }
}
```
run:
5
Something went wrong with your input!
java.lang.ArrayIndexOutOfBoundsException: 5
BUILD SUCCESSFUL (total time: 2 seconds)

run:
3
myarray at index 3 holds 4
BUILD SUCCESSFUL (total time: 1 second)
try{
    int[] myarray = {1,2,3,4};

    Scanner input=new Scanner(System.in);

    int k = input.nextInt();

    int numx = input.nextInt();

    int result = myarray[k] / numx;
}

} catch(ArrayIndexOutOfBoundsException aob){  //executed when k is 5 or more
    System.out.println("Something went wrong with your input, out of bound!");
} catch(ArithmeticException aex){  //executed when numx is zero
    System.out.println("you are dividing by zero!");
}
catch(Exception exc){  //general exception, catch any other error may occur
    System.out.println("Something went wrong with your input!");
}
Dealing with files

• Use Scanner class (we used before for user input), to read text files.
• Use PrintWriter class to write in a text file.
• We pass the “File” required for each of these classes.
• Scanner class offers two important functions:
  
1. `hasNextLine()`: True as long as the file has more lines to read.

2. `nextLine()`: returns the current line of the file as string.
Scanner input = new Scanner(System.in);

File file = new File("/Users/cebdevelopment/Downloads/testFile.txt");

input = new Scanner(file);

while (input.hasNextLine()) {
    String line = input.nextLine();
    System.out.println(line);
}

input.close();   //close the opened file
import java.io.*;  // this allows using Files and corresponding exceptions
import java.util.Scanner;
package cop2800;
import java.io.*;  // this allows using Files and corresponding exceptions
import java.util.Scanner;

public class mainclass {
    public static void main(String[] args) {
        try {
            Scanner input = new Scanner(System.in);

            File file = new File("/Users/cebdevelopment/Downloads/testFile.txt");
            input = new Scanner(file);

            while (input.hasNextLine()) {
                String line = input.nextLine();
                System.out.println(line);
            }
            input.close();  // close the opened file
        }
        catch (FileNotFoundException ex) {
            System.out.println("Can't locate this file !!");
        }
        catch (Exception ex) {
            System.out.println("Something went wrong !!");
        }
    }
}
try {
    File input = new File("/Users/cebdevelopment/Downloads/testFile_Input.txt");
    File output = new File("/Users/cebdevelopment/Downloads/testFile_Output.txt");
    Scanner sc = new Scanner(input); //Scanner to read
    PrintWriter printer = new PrintWriter(output); //PrintWriter to write
    while(sc.hasNextLine()) {
        String s = sc.nextLine();
        printer.write(s);
    }
    sc.close();
    printer.close();
}

catch(FileNotFoundException e) {
    System.err.println("File not found. Please scan in new file.");
}