CSE 1325

Week of 09/19/2022

Instructor : Donna French

Array List

When you write a program that collects inputs, you don't always know how many inputs you will have.

In this type of situation, an array list offers two significant advantages.

- Array lists can grow and shrink as needed
- The `ArrayList` class supplies methods for common tasks such as inserting and removing elements.

Declaring and Using `ArrayLists`

The following statement declares an array list of strings

```
ArrayList<String> names = new ArrayList<>();
```

The `ArrayList` is contained in a `java.util` package. In order to use array lists in your program, you need to use the statement

```
import java.util.ArrayList;
```

Declaring and Using `ArrayLists`

```
ArrayList<String> names = new ArrayList<>();
```

The type `ArrayList<String>` denotes an array list of `String` elements.

The angle brackets around the `String` type tell you that `String` is a template parameter.

You can replace `String` with any other **class** and get a different array list type.

For that reason, `ArrayList` is called a generic class.

Declaring and Using `ArrayLists`

```
ArrayList<CLASS> names = new ArrayList<>();
```

You CANNOT use primitive types as type parameters

there is no

`ArrayList<int>` or `ArrayList<double>`

Declaring and Using `ArrayLists`

```
ArrayList<CLASS> names = new ArrayList<>();
```

You can use other classes

```
ArrayList<String> SList = new ArrayList<>();
ArrayList<Integer> IList = new ArrayList<>();
ArrayList<Character> CList = new ArrayList<>();
ArrayList<Double> DList = new ArrayList<>();
```

Declaring and Using `ArrayLists`

```
ArrayList<CLASS> names = new ArrayList<>();
```

It is common error to forget the initialization with `new`

```
ArrayList<CLASS> names;
```

should be

```
ArrayList<CLASS> names = new ArrayList<>();
```

Declaring and Using `ArrayLists`

```
ArrayList<CLASS> names = new ArrayList<>();
```

Note the `()` after `new ArrayList<>` on the right hand side of the initialization.

This indicates that the constructor of the `ArrayList<CLASS>` is being called.

```
error: '(' or '[' expected ArrayList<String> names = new
ArrayList<>;
```

This line of code constructs an instance of object `ArrayList` named `names`.

Declaring and Using `ArrayLists`

```
ArrayList<CLASS> names = new ArrayList<>();
```

When the `ArrayList<`***`CLASS`***`>` is first constructed, it has size 0.

You use the `add` method to add an element to the end of the array list.

```
names.add("Happy");
names.add("Dopey");
names.add("Grumpy");
```

Declaring and Using `ArrayLists`

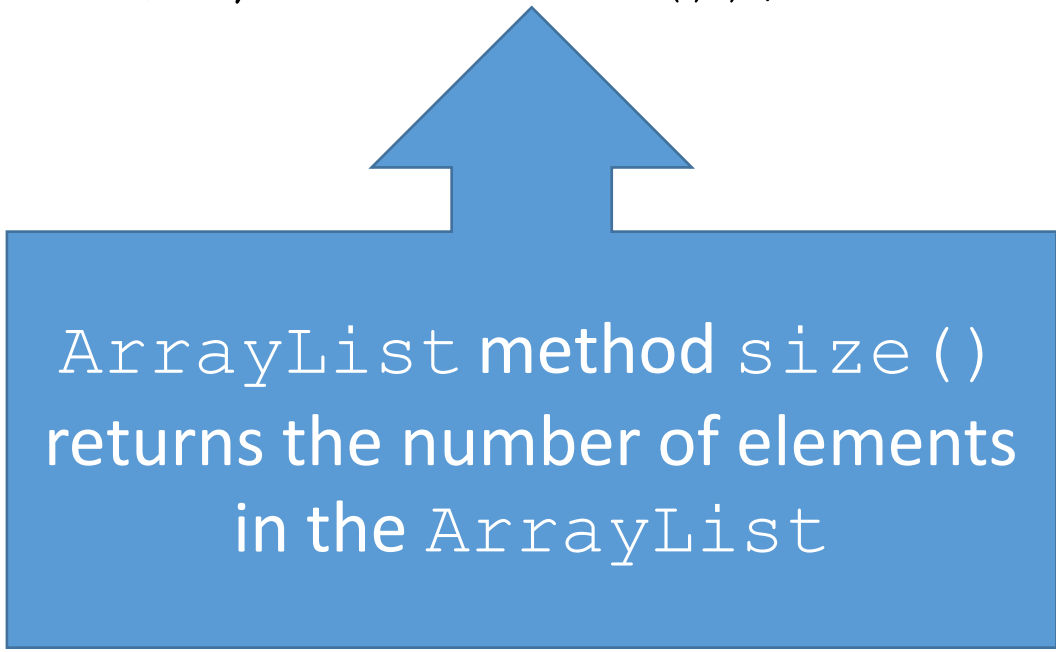
```
ArrayList<String> names = new ArrayList<>();
```

```
names.add("Happy");
names.add("Dopey");
names.add("Grumpy");
```

The size increases after each call to add. The `size` method yields the current size of the array list.

```
ArrayList<String> names = new ArrayList<>();

System.out.printf("ArrayList size BEFORE is %d\n", names.size());

names.add("Happy");
names.add("Dopey");
names.add("Grumpy");

System.out.printf("ArrayList size AFTER is %d\n", names.size());
```

```
ArrayList size BEFORE is 0
ArrayList size AFTER is 3
```



Declaring and Using `ArrayLists`

```
ArrayList<String> names = new ArrayList<>();
```

To print the contents of an array list, use the name of the array list

```
System.out.println(names);
```

Using just the array list name will print out the contents along with [].

```
[Happy, Dopey, Grumpy]
```

```
ArrayList<String> names = new ArrayList<>();

names.add("Happy");
names.add("Dopey");
names.add("Grumpy");

System.out.println(names);
```

```
[Happy, Dopey, Grumpy]
```

```
ArrayList<String> SList = new ArrayList<>();
ArrayList<Integer> IList = new ArrayList<>();
ArrayList<Character> CList = new ArrayList<>();
ArrayList<Double> DList = new ArrayList<>();

SList.add("Yarn");
IList.add(2);
CList.add('A');
DList.add(1.2);

System.out.println(SList);
System.out.println(IList);
System.out.println(CList);
System.out.println(DList);
```

```
[Yarn]
[2]
[A]
[1.2]
```

Declaring and Using `ArrayLists`

```
ArrayList<String> names = new ArrayList<>();
```

To obtain an array list element, use the `get` method, not the `[]` operator.

The first element is at index 0 just like arrays.

```
System.out.printf("%s", names.get(0));
String dwarfName =  names.get(1);
```

```
ArrayList<String> names = new ArrayList<>();

names.add("Happy");
names.add("Dopey");
names.add("Grumpy");

System.out.println(names.get(0));
String dwarfName = names.get(2);

System.out.println(dwarfName);
```

```
Happy
Grumpy
```

Declaring and Using `ArrayLists`

```
ArrayList<String> names = new ArrayList<>();
```

A very common bounds error is to use

```
int i = names.size();
String name = names.get(i);
```

```
25          names.add("Happy");
26          names.add("Dopey");
27          names.add("Grumpy");
28
29          int i = names.size();
            String name = names.get(i);
31
32      }
    }
34
```

studentcode.StudentCode > main > name

Output - StudentCode (run)

```
run:
Exception in thread "main" java.lang.IndexOutOfBoundsException: Index 3 out of b
ounds for length 3
        at java.base/jdk.internal.util.Preconditions.outOfBounds( Preconditions.j
ava:64)
        at java.base/jdk.internal.util.Preconditions.outOfBoundsCheckIndex( Preco
nditions.java:70)
        at java.base/jdk.internal.util.Preconditions.checkIndex( Preconditions.ja
va:248)
        at java.base/java.util.Objects.checkIndex( Objects.java:372)
        at java.base/java.util.ArrayList.get( ArrayList.java:458)
```

Declaring and Using `ArrayLists`

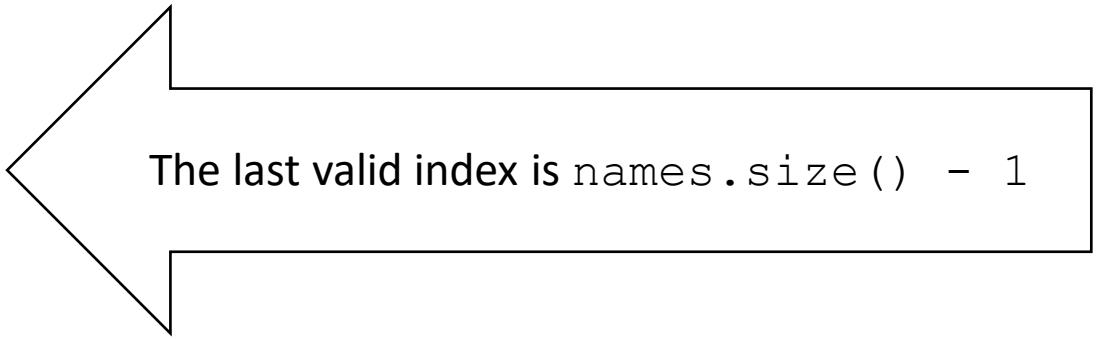
```
ArrayList<String> names = new ArrayList<>();
```

A very common bounds error is to use

```
int i = names.size();
String name = names.get(i);
```

instead use

```
int i = names.size() - 1;
String name = names.get(i);
```



Source History

```
25          names.add("Happy");
26          names.add("Dopey");
27          names.add("Grumpy");
28
29          int i = names.size() - 1;
30          String name = names.get(i);
31          System.out.println(name);
32
33      }
}
```

studentcode.StudentCode > main

Output - StudentCode (run)

```
r.properties
deps-jar:
Updating property file: C:\Users\Donna\Desktop\UTA\CSE1310\Programs\StudentCode\build\built-jar.properties
Compiling 1 source file to C:\Users\Donna\Desktop\UTA\CSE1310\Programs\StudentCode\build\classes
compile:
run:
Grumpy
BUILD SUCCESSFUL (total time: 1 second)
```

Declaring and Using `ArrayLists`

```
ArrayList<String> names = new ArrayList<>();
```

A `for` loop can be used to display all elements of an array list

```
for (int i = 0; i < names.size(); i++)
{
   System.out.println(names.get(i));
}
```

Using a for loop rather than just printing the array list give us more control over the formatting and what we need to do with the data.

```
ArrayList<String> names = new ArrayList<>();

names.add("Happy");
names.add("Dopey");
names.add("Grumpy");

for (int i = 0; i < names.size(); i++)
{
     System.out.println(names.get(i));
}
```

```
Happy
Dopey
Grumpy
```

```
package arraylistdemo;

import java.util.ArrayList;

public class ArrayListDemo
{
    public static void main(String[] args)
    {
        ArrayList<Integer> IList = new ArrayList<>();

        IList.add(583);
        IList.add(274);
        IList.add(987);
        IList.add(295);

        for (int i = 0; i < IList.size(); i++)
        {
            System.out.printf("%d\t", IList.get(i));
        }
        System.out.println();
    }
}
```

583 274 987 295

Declaring and Using `ArrayLists`

```
ArrayList<String> names = new ArrayList<>();
```

A `for` loop can also be used to add elements to an array list

```
for (int i = 0; i < 26; i++)
{
   String myLetter = String.valueOf((char)(i+65));
   alphabet.add(i, myLetter);
}
System.out.println(alphabet);
```

```
[A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z]
```

Declaring and Using `ArrayLists`

```
ArrayList<String> names = new ArrayList<>();
```

To set an array list element to a new value, use the `set` method

```
names.set(2, "Doc");
```

This call sets position 2 of the `names` array list to "`Doc`" which overwrites whatever value was there before.

```
ArrayList<String> names = new ArrayList<>();

names.add("Happy");
names.add("Dopey");
names.add("Grumpy");

int i = names.size() - 1;
String name = names.get(i);
System.out.println(name);
names.set(i, "Doc");
System.out.println(names.get(i));
```

```
Grumpy
Doc
```

Declaring and Using `ArrayLists`

```
ArrayList<String> names = new ArrayList<>();
```

You can insert an element in the middle of an array list.

```
names.add(1, "Sleepy");
```

This adds a new element at position 1 and moves all elements with index 1 or larger by one position. After each call to the `add` method, the size of the array list increases by 1.

```
ArrayList<String> names = new ArrayList<>();

names.add("Happy");
names.add("Dopey");
names.add("Grumpy");

System.out.printf("%s is size %d\n", names, names.size());

names.add(1, "Sleepy");

System.out.printf("%s is size %d\n", names, names.size());
```

```
[Happy, Dopey, Grumpy] is size 3
[Happy, Sleepy, Dopey, Grumpy] is size 4
```

Declaring and Using `ArrayLists`

```
ArrayList<String> names = new ArrayList<>();
```

You can remove an element from anywhere in an array list.

```
names.remove(1);
```

This removes the element at position 1 and moves all elements with index 1 or larger by one position. After each call to the `remove` method, the size of the array list decreases by 1.

```
ArrayList<String> names = new ArrayList<>();

names.add("Happy");
names.add("Dopey");
names.add("Grumpy");

System.out.printf("%s is size %d\n", names, names.size());

names.add(1, "Sleepy");
System.out.printf("%s is size %d\n", names, names.size());
names.remove(2);
System.out.printf("%s is size %d\n", names, names.size());
```

```
[Happy, Dopey, Grumpy] is size 3
[Happy, Sleepy, Dopey, Grumpy] is size 4
[Happy, Sleepy, Grumpy] is size 3
```

Array List Summary

```
ArrayList<String> myArrList = new ArrayList<>();
```

`myArrList.add("Sneezy");`	adds elements to end
`System.out.print(myArrList);`	prints `myArrList` with []
`myArrList.add(1, "Bashful");`	inserts element at index 1
`myArrList.remove(0);`	removes element at index 0
`myArrList.set(2, "Silly");`	replaces element at 2
`String myName = myArrList.get(i)`	gets an element

```
public static void GetTheGangTogether(ArrayList<String> SG)
{
	SG.add("Happy");
	SG.add("Dopey");
	SG.add("Doc");
	SG.add("Bashful");
	SG.add("Dreamy");
	SG.add("Sneezy");
	SG.add("Sleepy");
	SG.add("Stealthy");
}

public static void main(String[] args)
{
	ArrayList<String> SnowGang = new ArrayList<>();

	GetTheGangTogether(SnowGang);

	System.out.println(SnowGang);
}
```

```
[Happy, Dopey, Doc, Bashful, Dreamy, Sneezy, Sleepy, Stealthy]
```

[Happy, Dopey, Doc, Bashful, Dreamy, Sneezy, Sleepy, Stealthy]

```
SnowGang.add(0, "Snow White");
SnowGang.set(5, "Grumpy");
SnowGang.remove(SnowGang.size() - 1);

System.out.println(SnowGang);
```

[Snow White, Happy, Dopey, Doc, Bashful, Grumpy, Sneezy, Sleepy]

Sorting
Arrays vs ArrayLists

To sort an array

```
Arrays.sort(array_name);
```

To sort an array list

```
Collections.sort(ArrayList_Name);
```

Sorting

To sort an `ArrayList`

```
Collections.sort(ArrayListName);
```

To sort an `ArrayList` in reverse order

```
Collections.sort(ArrayListName, Collections.reverseOrder);
```

Converting `ArrayList` to `String`

To convert an `ArrayList` to string

```
ArrayList<Integer> HelloList = new ArrayList<>();
HelloList.add(9399);
HelloList.add(1234);
HelloList.add(3450);

String HelloListString = HelloList + "";

System.out.println(HelloList);
System.out.println(HelloListString);
```

```
[9399, 1234, 3450]
[9399, 1234, 3450]
```

The + has the ability to convert the `ArrayList` to a `String` by concatenated it with the `String` ""

Converting to String

To convert an array to a string

```
int Hello[] = {1,2,3,4,5};
String Bye = Arrays.toString(Hello);
System.out.println(Bye);
```

```
[1, 2, 3, 4, 5]
```

```
int Hello[] = {1,2,3,4,5};
String Bye = Hello + "";

[C@7b23ec81
```

references to the array – not the array

```
char Hello[] = {'A','B','C'};
String Bye = Hello + "";

[I@7b23ec81
```

Printing

To print an `ArrayList`

```
System.out.print(ArrayListName);
```

To print an array

```
System.out.print(Arrays.toString(ArrayName));
```

Copying
Arrays vs ArrayLists

To copy an Array

```
NewArray = Arrays.copyOf(OldArray, length of NewArray);
```

To copy an ArrayList

```
NewArrayList = new ArrayList<String>(OldArrayList);
```

Array Copy

```
NewArray = Arrays.copyOf(OldArray, length of NewArray);


int Hello[] = {1,2,3,4,5};

int Hola[] = Arrays.copyOf(Hello, Hello.length*4);

System.out.print(Arrays.toString(Hola));


[1, 2, 3, 4, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

ArrayList Copy

```
NewArrayList = new ArrayList<>(OldArrayList);
```

```
ArrayList<Integer> AdiosList = new ArrayList<>(HelloList);
```

```
System.out.println(AdiosList + "");
```

```
[9399, 1234, 3450]
```

```
int Hola[] = Arrays.copyOf(Hello, Hello.length*4);

System.out.printf("Printing Hello : %s\n", Arrays.toString(Hello));
System.out.printf("Printing Hola  : %s\n", Arrays.toString(Hola));

Hola[1] = -1;

System.out.printf("Printing Hello : %s\n", Arrays.toString(Hello));
System.out.printf("Printing Hola  : %s\n", Arrays.toString(Hola));
```

```
Printing Hello : [1, 2, 3, 4, 5]
Printing Hola  : [1, 2, 3, 4, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Printing Hello : [1, 2, 3, 4, 5]
Printing Hola  : [1, -1, 3, 4, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

Array Copy

```
int [] Hello = {1,2,3,4,5};
int [] Hola = Hello;

Hola[1] = -1;

System.out.printf("Printing Hello : %s\n", Arrays.toString(Hello));
System.out.printf("Printing Hola  : %s\n", Arrays.toString(Hola));
```

```
Printing Hello : [1, -1, 3, 4, 5]
Printing Hola  : [1, -1, 3, 4, 5]
```

How to Convert a `char` to a `String`

A variable of type `char` cannot be assigned to a variable of type `String`.

```
char Letter = 'A';
String Word = Letter;
```

```
error: incompatible types: char cannot be converted to
String
        String Word = Letter;
```

How to Convert a `char` to a `String`

```
char Letter = 'A';
String Word = (String)Letter;
```

A variable of type `char` cannot be cast to a type `String` either.

`char` is a primitive type while `String` is an object (`char` is lowercase – most primitives are and `String` is uppercase – most objects are).

Objects have methods – you can ask a `String` how long it is but you cannot ask a `char`.

How to Convert a `char` to a `String`

```
char Letter = 'A';
String Word = Letter + "";
```

Just as we saw with `ArrayList` printing, the + is able to convert `Letter` to a `String` because it needs to concatenate it with the empty `String` "".

```
System.out.print(Letter);
System.out.print(Word);
```

```
AA
```

How to Convert an `int` to a `String`

A variable of type `int` cannot be assigned to a variable of type `String`.

```
int Number = 1;
String StringNumber = Number;
```

```
error: incompatible types: int cannot be converted to
String
        String StringNumber = Number
```

How to Convert an `int` to a `String`

```
int Number = 1;
String StringNumber = (String)Number;
```

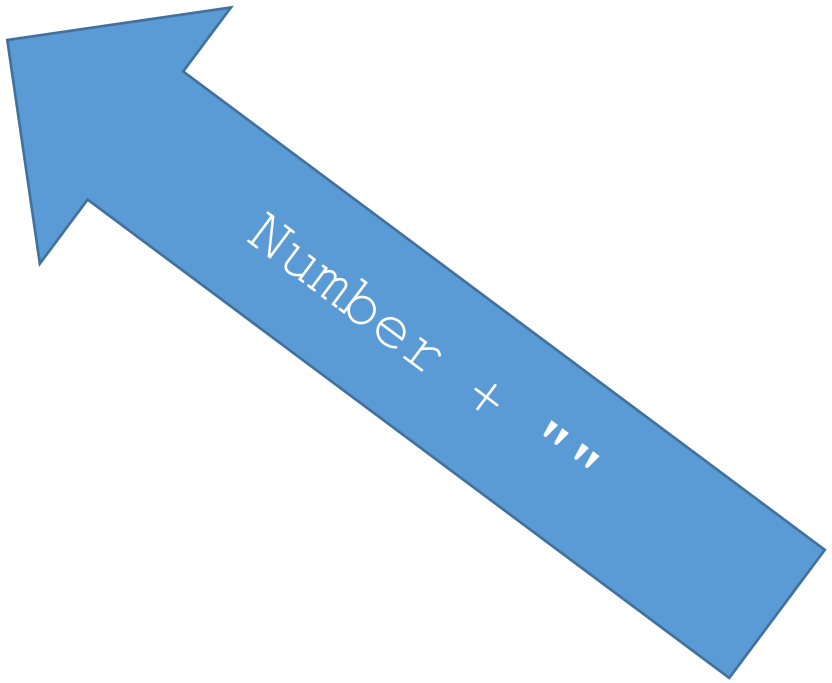
A variable of type `int` cannot be cast to a type `String` either.

`int` is a primitive type while `String` is an object (`int` is lowercase – most primitives are and `String` is uppercase – most objects are).

```
error: incompatible types: int cannot be converted to
String
        String StringNumber = (String)Number
```

How to Convert an `int` to a `String`

```
int Number = 1;
String StringNumber1 = String.valueOf(Number);
String StringNumber2 = Integer.toString(Number);
String StringNumber3 = "" + Number;

System.out.print(Number);
System.out.print(StringNumber1);
System.out.print(StringNumber2);
System.out.print(StringNumber3);

1111
```



Array List Summary

```
ArrayList<String> myArrList = new ArrayList<>();
```

`myArrList.add("Sneezy");`	adds elements to end
`myArrList.add(1, "Bashful");`	inserts element at index 1
`System.out.print(myArrList);`	prints `myArrList` with []
`myArrList.remove(0);`	removes element at index 0
`myArrList.set(2, "Silly");`	replaces element at 2
`String myName = myArrList.get(i)`	gets an element

`ArrayList` Exercise

- Declare an `ArrayList` named `myShoppingList` of type `String`
  - Which `import` needs to be added?
- Add `bread` to your shopping list
- Add `milk` to your shopping list at position 1
- Remove `bread` from your shopping list
- Replace the first item in your shopping list with `cheese`
- Add `bread` to your shopping list
- Change `cheese` to `butter` in your shopping list
- Create and initialize `String myItem` to the last element of `myShoppingList`

```
import java.util.ArrayList;

public class StudentCode
{
    public static void main(String[] args)
    {
        ArrayList<String> myShoppingList = new ArrayList<>();

        myShoppingList.add("bread");
        myShoppingList.add(1, "milk");
        myShoppingList.remove(0);
        myShoppingList.set(0, "cheese");
        myShoppingList.add("bread");
        myShoppingList.set(0, "butter");
        String myItem = myShoppingList.get(myShoppingList.size()-1);

        System.out.print(myShoppingList);
        System.out.println(myItem);
    }
}
```

Exception Handling

There are two aspects to dealing with program errors: detection and handling.

Some problems you can detect and prevent.

For example…

You are detecting when the user enters a choice that is not within range of your menu options.

```
Please choose from the following options


0. No pencils for me today
1. Purchase pencils
2. Check inventory level
3. Check change level

Choice : 4

Invalid menu choice.  Please choose again.

Please choose from the following options


0. No pencils for me today
1. Purchase pencils
2. Check inventory level
3. Check change level

Choice :
```

Exception Handling

Some errors we cannot detect and prevent. When this situation occurs, we need exception handling.

For example…

A letter is entered for a menu choice...

You may have noticed that if you enter a value like "a" instead of number, something happens….

```
0. No pencils for me today
1. Purchase pencils
2. Check inventory level
3. Check change level

Choice : a
Exception in thread "main" java.util.InputMismatchException
      at java.base/java.util.Scanner.throwFor(Scanner.java:939)
      at java.base/java.util.Scanner.next(Scanner.java:1594)
      at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
      at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
      at
code1_1000074079.Code1_1000074079.PencilMenu(Code1_1000074079.java:35)
      at code1_1000074079.Code1_1000074079.main(Code1_1000074079.java:106)
C:\Users\frenc\Documents\NetBeansProjects\Code1_1000074079\nbproject\build-
impl.xml:1355: The following error occurred while executing this line:
C:\Users\frenc\Documents\NetBeansProjects\Code1_1000074079\nbproject\build-
impl.xml:961: Java returned: 1
BUILD FAILED (total time: 1 minute 6 seconds)
```

```
35   choice = in.nextInt();
106  menu_choice = PencilMenu();
```

Exception Handling

To handle exceptions like this, we need to add exception handling to our program.

Exception handling allows us to handle the error rather than just letting our program explode.

```
0. No pencils for me today
1. Purchase pencils
2. Check inventory level
3. Check change level

Choice : a
Exception in thread "main" java.util.InputMismatchException
      at java.base/java.util.Scanner.throwFor(Scanner.java:939)
      at java.base/java.util.Scanner.next(Scanner.java:1594)
      at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
      at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
      at
code1_1000074079.Code1_1000074079.PencilMenu(Code1_1000074079.java:35)
      at code1_1000074079.Code1_1000074079.main(Code1_1000074079.java:106)
C:\Users\frenc\Documents\NetBeansProjects\Code1_1000074079\nbproject\build-
impl.xml:1355: The following error occurred while executing this line:
C:\Users\frenc\Documents\NetBeansProjects\Code1_1000074079\nbproject\build-
impl.xml:961: Java returned: 1
BUILD FAILED (total time: 1 minute 6 seconds)
```

```
35   choice = in.nextInt();
106  menu_choice = PencilMenu();
```

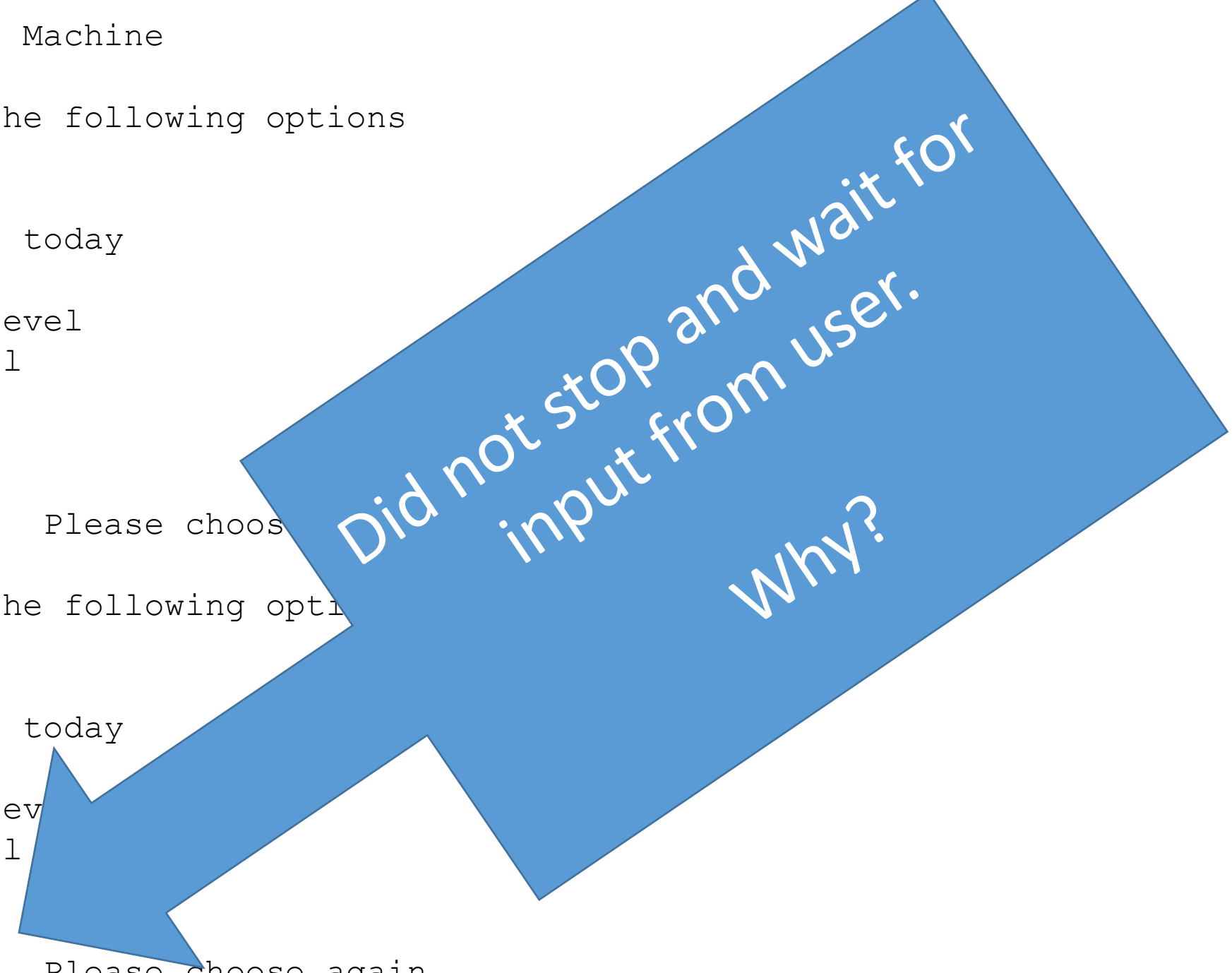
```
try
{
    choice = in.nextInt();
}
catch (Exception e)
{
    System.out.println("Invalid menu choice.  Please choose again.\n");
}
```

```
Welcome to my Pencil Machine

Please choose from the following options


0. No pencils for me today
1. Purchase pencils
2. Check inventory level
3. Check change level

Choice : a
Invalid menu choice.  Please choose again.

BUILD SUCCESSFUL (total time: 4 seconds)
```

`choice` was initialized to `0` so when the exception happened, `choice` was left at `0` and `0` is an acceptable value – program completes when `0` is entered.

```
try
{
    choice = in.nextInt();
}
catch (Exception e)
{
    choice = -1;
}
```

```
Welcome to my Pencil Machine

Please choose from the following options


0. No pencils for me today
1. Purchase pencils
2. Check inventory level
3. Check change level

Choice : a

Invalid menu choice.  Please choos

Please choose from the following opti


0. No pencils for me today
1. Purchase pencils
2. Check inventory lev
3. Check change level

Choice :
Invalid menu choice.  Please choose again.
```

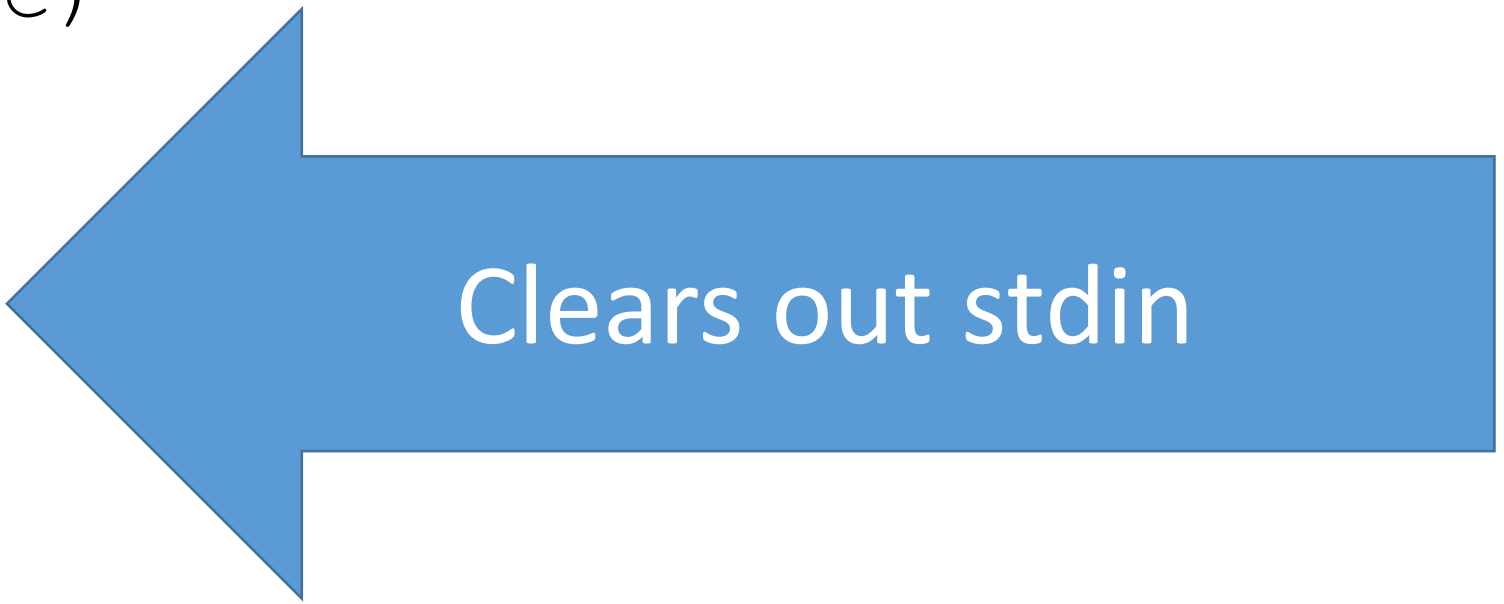


```
Scanner in = new Scanner(System.in);
System.out.print("Enter something ");
int choice = 0;

try
{
    choice = in.nextInt();
    System.out.printf("You entered %d\n", choice);
}
catch (Exception e)
{
    System.out.printf("You entered %s", in.next());
}
```

```
Enter something 1
You entered 1

Enter something a
You entered a

Enter something dog cat bird
You entered dog
```

```
Scanner in = new Scanner(System.in);
System.out.print("Enter something ");
int choice = 0;

try
{
    choice = in.nextInt();
    System.out.printf("You entered %d\n", choice);
}
catch (Exception e)
{
    System.out.printf("You entered %s", in.nextLine());
}
```

```
try
{
    choice = in.nextInt();
}
catch (Exception e)
{
    choice = -1;
    in.nextLine();
}
```



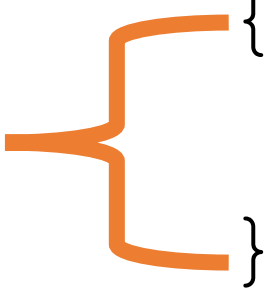
```
Welcome to my Pencil Machine

Please choose from the following options


0. No pencils for me today
1. Purchase pencils
2. Check inventory level
3. Check change level

Choice : lajf alkfj;as alfkjadlkfj

Invalid menu choice.  Please choose again.
```

```
Please choose from the following options


0. No pencils for me today
1. Purchase pencils
2. Check inventory level
3. Check change level

Choice : 3

The current change level is $5.00

Please choose from the following options


0. No pencils for me today
1. Purchase pencils
2. Check inventory level
3. Check change level

Choice :
```

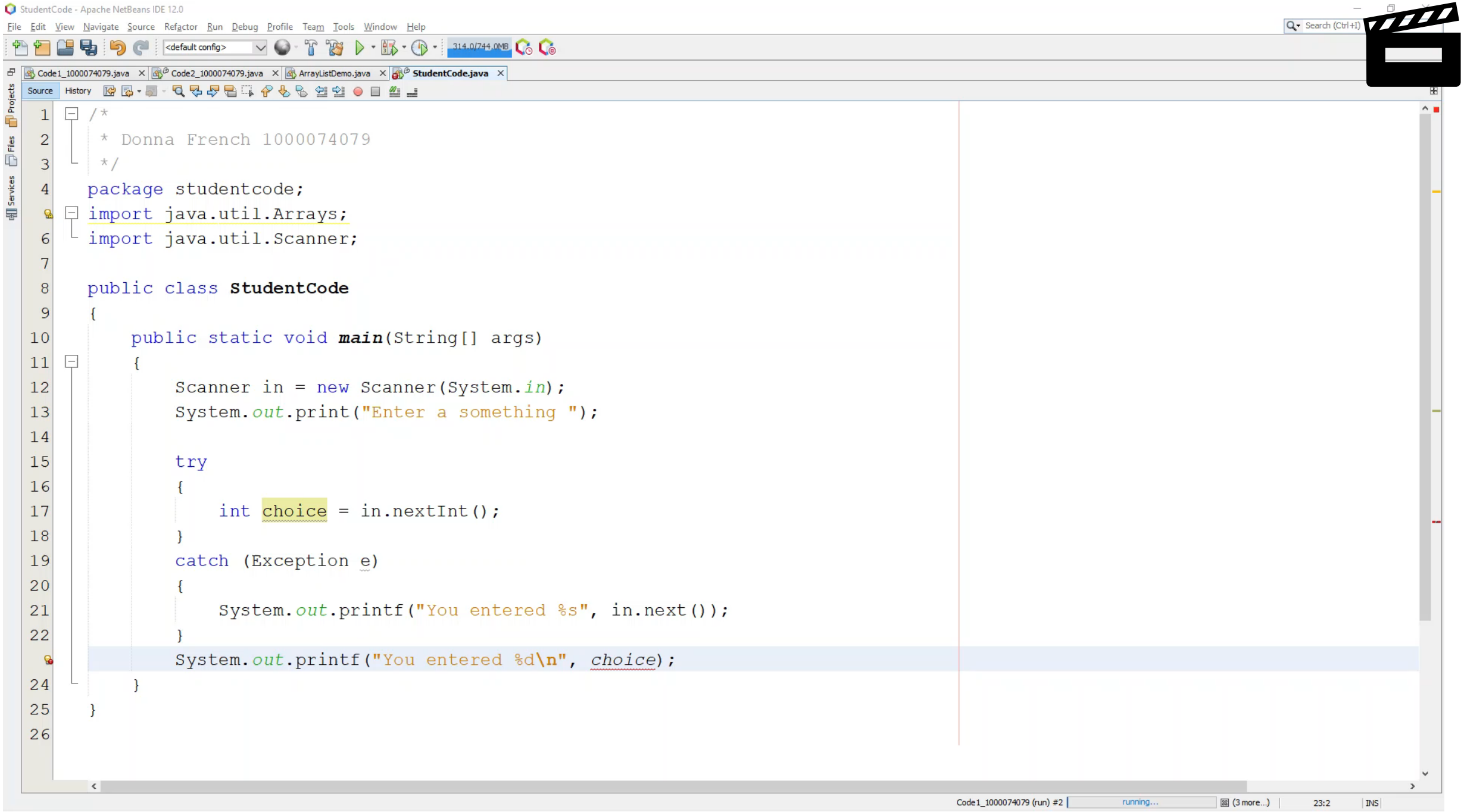
```
Scanner in = new Scanner(System.in);
System.out.print("Enter something ");
int choice = 0;

try
{
   int choice = in.nextInt());
   System.out.printf("You entered %d\n", choice);
}
catch (Exception e)
{
   System.out.printf("You entered %s", in.nextLine());
}
System.out.printf("You entered %d\n", choice);
```

```
C:\Users\frenc\Documents\NetBeansProjects\StudentCode\src\studentcode\Studen
tCode.java:23: error: cannot find symbol
        System.out.printf("You entered %d\n", choice);
  symbol:   variable choice
  location: class StudentCode
1 error
```

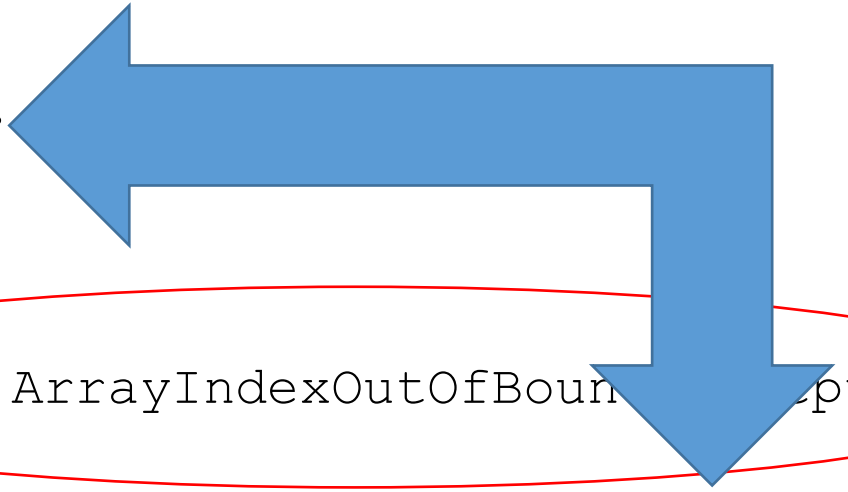
This is a block – any variable declared inside the block can only be seen INSIDE the block.



```
try
{
    int choice = in.nextInt();
}
```

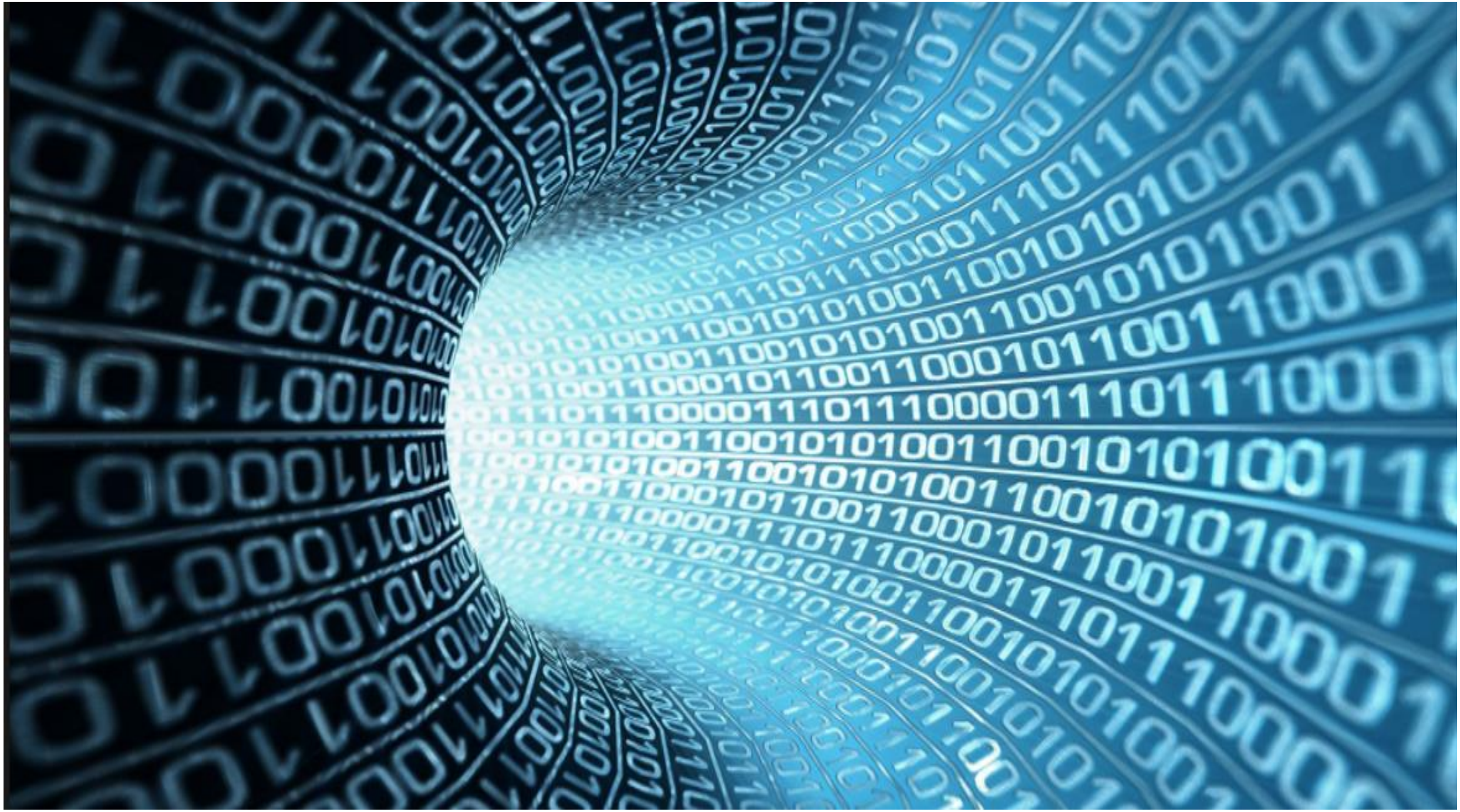

StudentCode - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
314.0/744.0MB
Code1_1000074079.java
Code2_1000074079.java
ArrayListDemo.java
StudentCode.java
Source
History
Projects
Files
Services
1 /*
2 * Donna French 1000074079
3 */
4 package studentcode;
5 import java.util.Arrays;
6 import java.util.Scanner;
7
8 public class StudentCode
9 {
10 public static void main(String[] args)
11 {
12 Scanner in = new Scanner(System.in);
13 System.out.print("Enter a something ");
14
15 try
16 {
17 int choice = in.nextInt();
18 }
19 catch (Exception e)
20 {
21 System.out.printf("You entered %s", in.next());
22 }
System.out.printf("You entered %d\n", choice);
24 }
25 }
26
Code1_1000074079 (run) #2
running...
(3 more...)
23:2
INS

```
Scanner in = new Scanner(System.in);

int [] Seal = {1,2,3,4,5};

for (int i = 0; i <= Seal.length; i++)
{
     System.out.print(Seal[i]);
}
```



```
Exception in thread "main" java.lang.ArrayIndexOutOfBoun        ption:
Index 5 out of bounds for length 5
12345 at trycatchdemo.TrycatchDemo.main(TrycatchDemo.java:17)
C:\Users\frenc\Documents\NetBeansProjects\trycatchDemo\nbproject\build-
impl.xml:1355: The following error occurred while executing this line:
C:\Users\frenc\Documents\NetBeansProjects\trycatchDemo\nbproject\build-
impl.xml:961: Java returned: 1
BUILD FAILED (total time: 1 second)
```

```
Scanner in = new Scanner(System.in);

int Seal[] = {1,2,3,4,5};

for (int i = 0; i <= Seal.length; i++)
{
   try
   {
      System.out.print(Seal[i]);
   }
   catch (Exception Team)
   {
      System.out.printf("\n%d is a bad index\n", i);
   }
}
```



File Handling

When reading from standard input (the keyboard), we created a Scanner object

```
Scanner in = new Scanner(System.in);
```

To read from a file, we need to create a `File` handle. A handle is how we refer to the variable that connects the file with our program.

```
File fileIn = new File("input.txt");
```

and then we can open a `Scanner` using that file handle instead of `stdin/System.in`

```
Scanner inFileRead = new Scanner(fileIn);
```

File Handling

Using class `File` to create a file handle

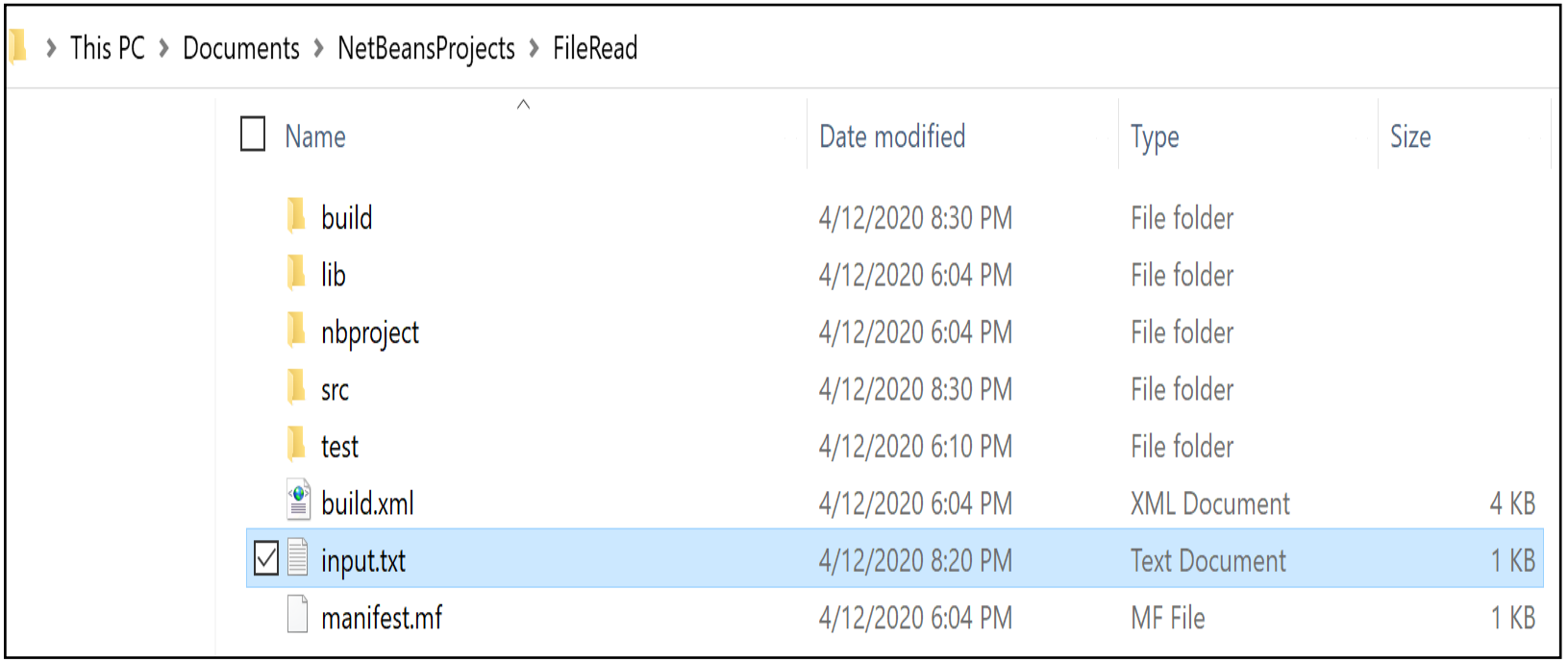
```
File fileIn = new File("input.txt");
```

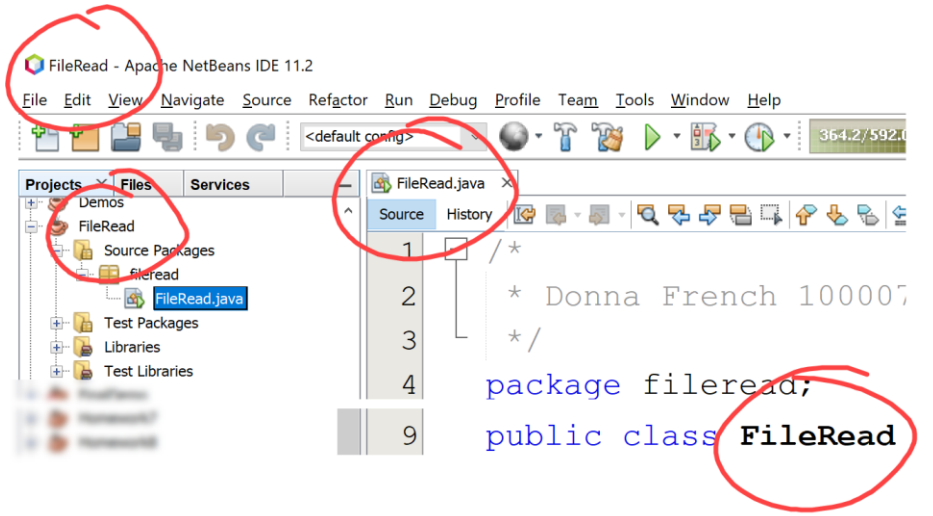
requires a new import

```
import java.io.File
```

File Handling

When you open a file from inside your NetBeans Java program, it will default to looking for the file in the project folder. Any files you need to read will need to be in that folder unless you add a path.





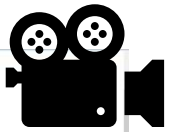
File Handling

You can add a path to a different (non default) location

```
File fileIn = new File("input.txt");
Scanner inFileRead = new Scanner(fileIn);
```

If you give a path to the file, then you must use \\ instead of slash in order to escape the \ and force it to be recognized as a \ and not an escape character.

```
File fileIn = new File("C:\\temp\\input.txt");
Scanner inFileRead = new Scanner(fileIn);
```



```java
/*
 * Donna French 1000074079
 */
package fileread;
import java.util.Scanner;
import java.io.File;

public class FileRead
{
    public static void main(String[] args)
    {
        File fileIn = new File("input.txt");
        Scanner inFileRead = new Scanner(fileIn);



    }
}
```

File Handling

When using `Scanner` to open a file

```
File fileIn = new File("input.txt");
Scanner inFileRead = new Scanner(fileIn);
```

if the file does not exist, it will be created but `Scanner` will throw an exception and we have to add code to handle that situation

For now, simply add this to your `main` method

```
public static void main(String[] args) throws FileNotFoundException
```

File Handling

When your program opens a file

```
File fileIn = new File("input.txt");
Scanner inFileRead = new Scanner(fileIn);
```

your program should also close the file

```
inFileRead.close()
```

File Handling

Your program can prompt for a filename

```
System.out.print("Enter file name ");
String FileName = in.next();
```

and then use that filename to open the file

```
File fileIn = new File(FileName);
Scanner inFileRead = new Scanner(fileIn);
```

File Handling

The \\ only needs to be used when the filename is hardcoded in the program. If you are entering the filename at a prompt, the \\ are not needed.

```
Scanner in = new Scanner(System.in);
System.out.print("Enter a file name ");
String FileName = in.next();

File fileIn = new File(FileName);
Scanner inFileRead = new Scanner(fileIn);

Enter file name C:\temp\input.txt
```

File Handling

Once your program opens a file

```
File fileIn = new File(FileName);
Scanner inFileRead = new Scanner(fileIn);
```

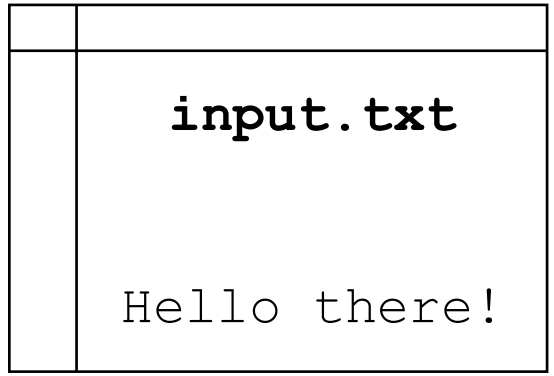
your program can read from it using the same input methods we have been using for keyboard input

```
String FileLine = inFileRead.nextLine();
```

File Handling

Once the file is open, we can use our familiar input methods to read from it just like we read from our keyboard version of Scanner.

When reading from a file, the same rules apply.



If we use

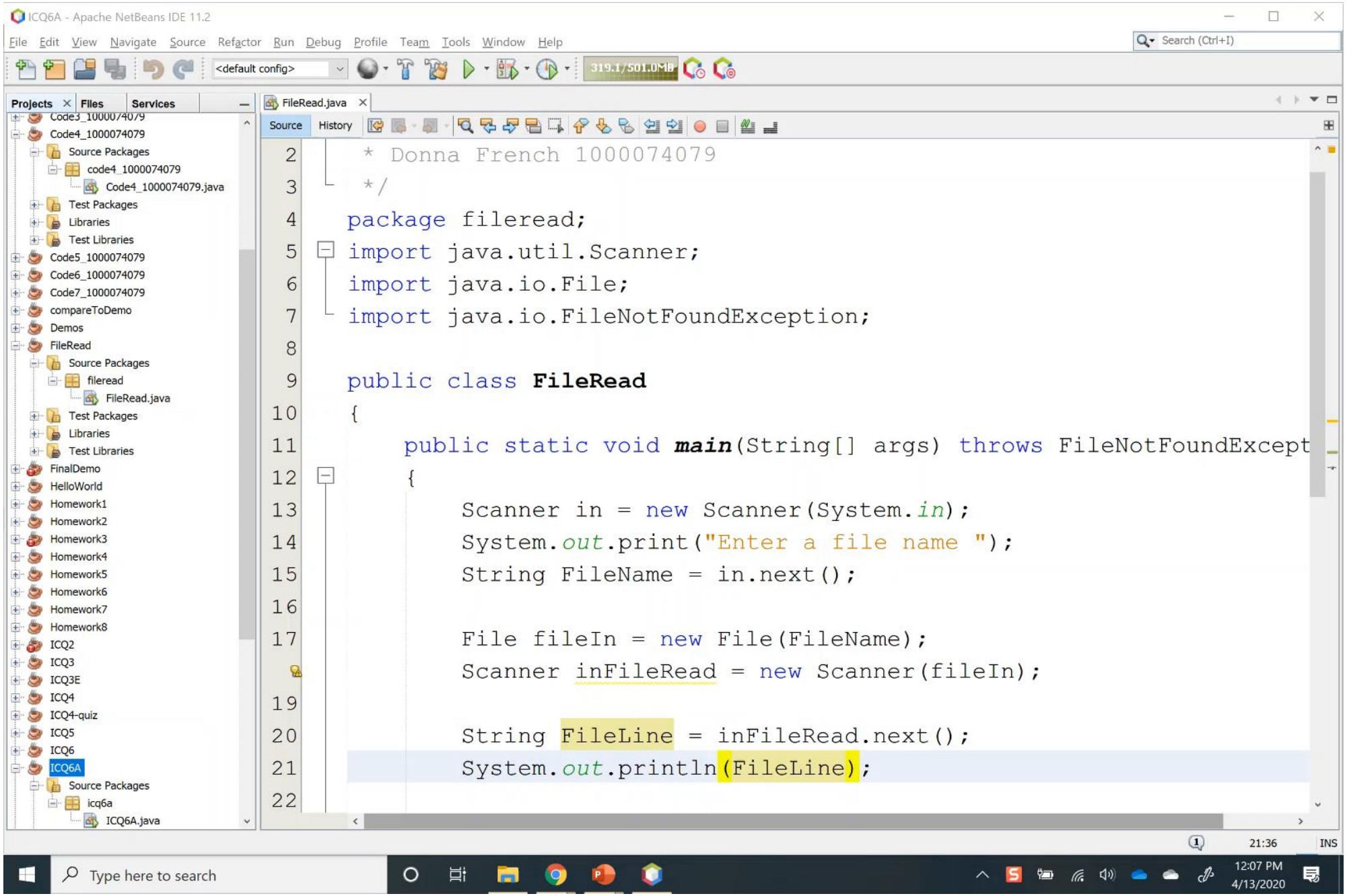
```
String FileLine = inFileRead.next();
```

Then

```
System.out.println(FileLine);
```

will print just

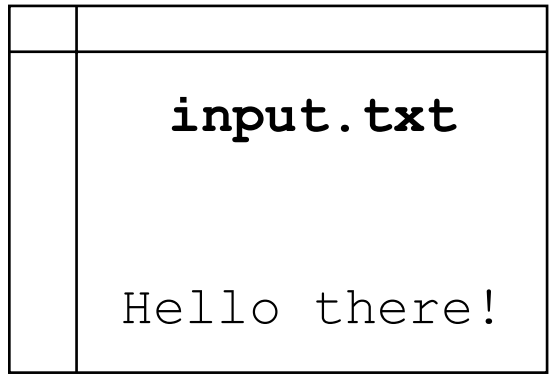
```
Hello
```


ICQ6A - Apache NetBeans IDE 11.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
319.1/501.0MB
Projects Files Services
Code3_1000074079
Code4_1000074079
Source Packages
code4_1000074079
Code4_1000074079.java
Test Packages
Libraries
Test Libraries
Code5_1000074079
Code6_1000074079
Code7_1000074079
compareToDemo
Demos
FileRead
Source Packages
fileread
FileRead.java
Test Packages
Libraries
Test Libraries
FinalDemo
HelloWorld
Homework1
Homework2
Homework3
Homework4
Homework5
Homework6
Homework7
Homework8
ICQ2
ICQ3
ICQ3E
ICQ4
ICQ4-quiz
ICQ5
ICQ6
ICQ6A
Source Packages
icq6a
ICQ6A.java
FileRead.java
Source History
* Donna French 1000074079
*/
package fileread;
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;
public class FileRead
{
public static void main(String[] args) throws FileNotFoundExcept
{
Scanner in = new Scanner(System.in);
System.out.print("Enter a file name ");
String FileName = in.next();
File fileIn = new File(FileName);
Scanner inFileRead = new Scanner(fileIn);
String FileLine = inFileRead.next();
System.out.println(FileLine);
21:36
INS
Type here to search
12:07 PM
4/13/2020

File Handling

If we use

```
String fileLine = inFileRead.nextLine();
```

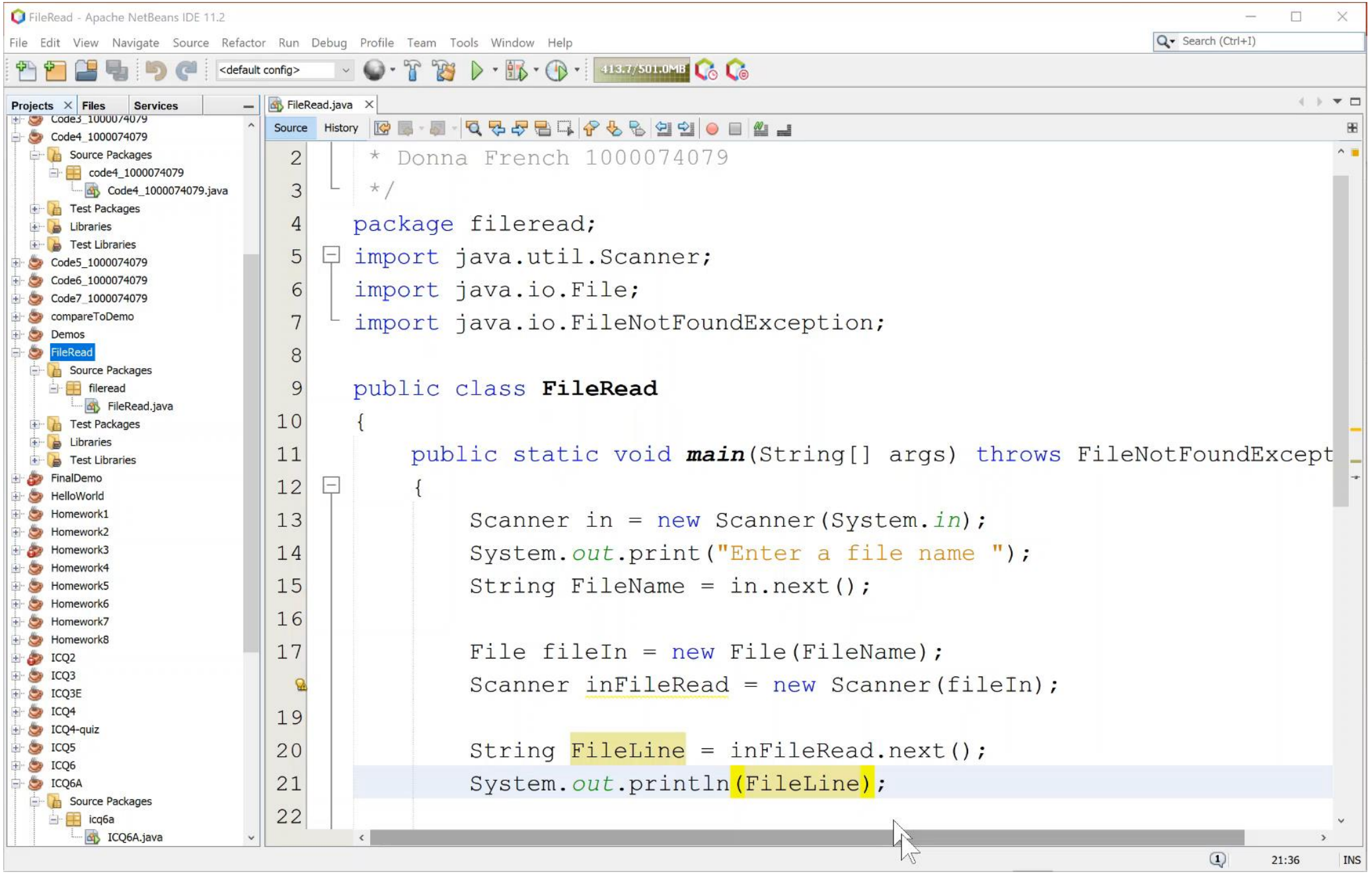


Then

```
System.out.println(fileLine);
```

will print the whole line from the file

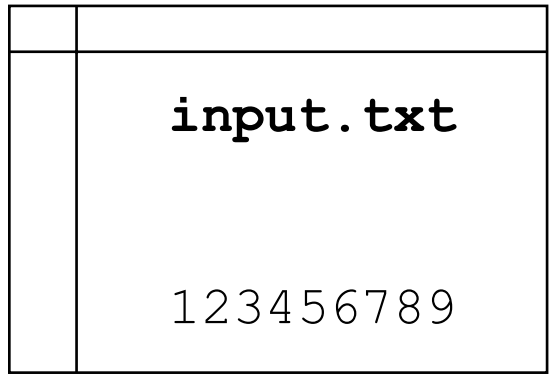
```
Hello there!
```


FileRead - Apache NetBeans IDE 11.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
413.7/501.0MB
Projects
Files
Services
Code4_1000074079
Source Packages
code4_1000074079
Code4_1000074079.java
Test Packages
Libraries
Test Libraries
Code5_1000074079
Code6_1000074079
Code7_1000074079
compareToDemo
Demos
FileRead
Source Packages
fileread
FileRead.java
Test Packages
Libraries
Test Libraries
FinalDemo
HelloWorld
Homework1
Homework2
Homework3
Homework4
Homework5
Homework6
Homework7
Homework8
ICQ2
ICQ3
ICQ3E
ICQ4
ICQ4-quiz
ICQ5
ICQ6
ICQ6A
Source Packages
icq6a
ICQ6A.java
FileRead.java
Source
History
* Donna French 1000074079
*/
package fileread;
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;
public class FileRead
{
public static void main(String[] args) throws FileNotFoundExcept
{
Scanner in = new Scanner(System.in);
System.out.print("Enter a file name ");
String FileName = in.next();
File fileIn = new File(FileName);
Scanner inFileRead = new Scanner(fileIn);
String FileLine = inFileRead.next();
System.out.println(FileLine);
21:36
INS

File Handling

We can read the file as numbers (if the file contains numbers)

```
int FileLine = inFileRead.nextInt();
```

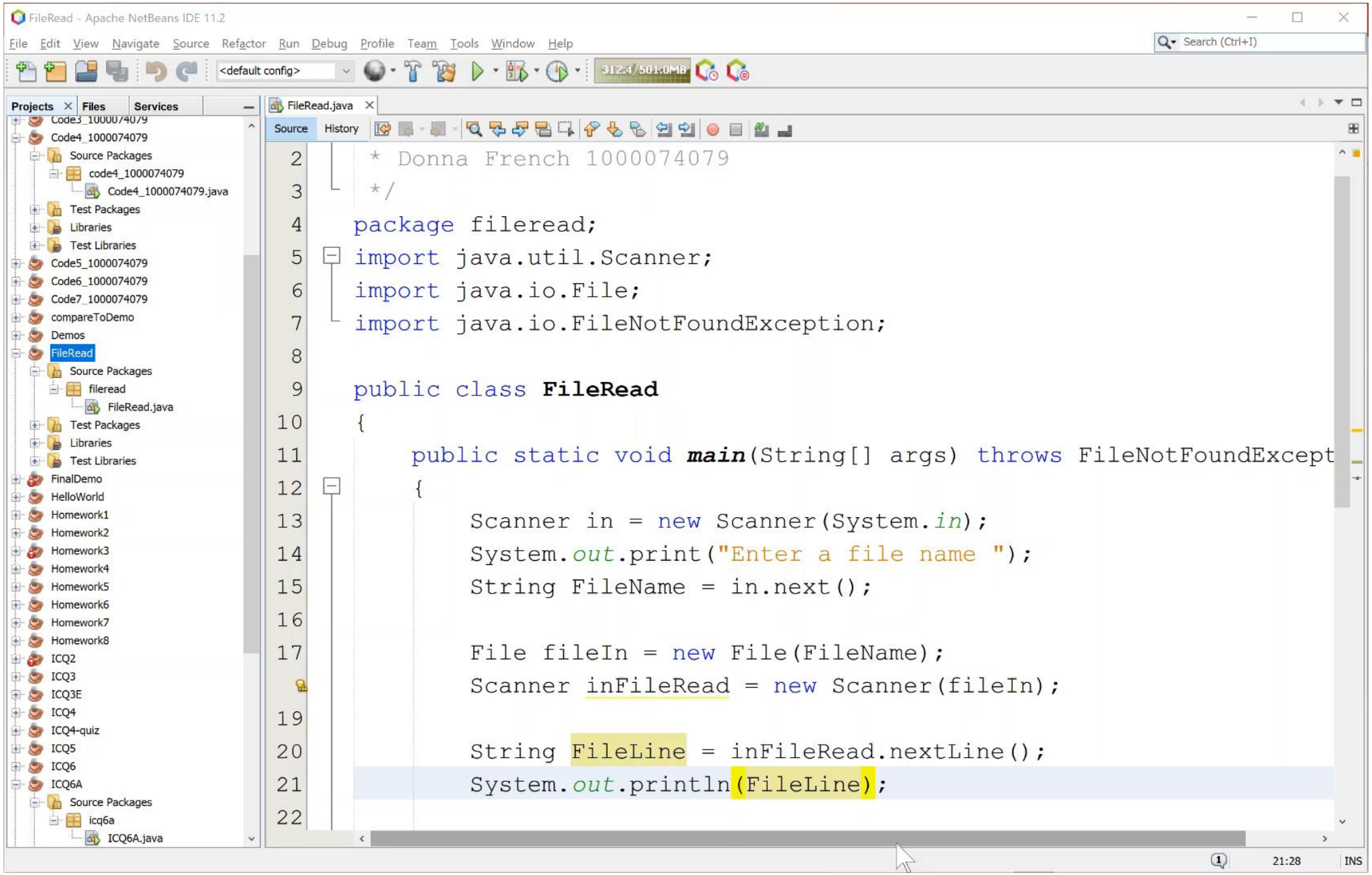


Then

```
System.out.println(FileLine);
```

will print the whole number from the file

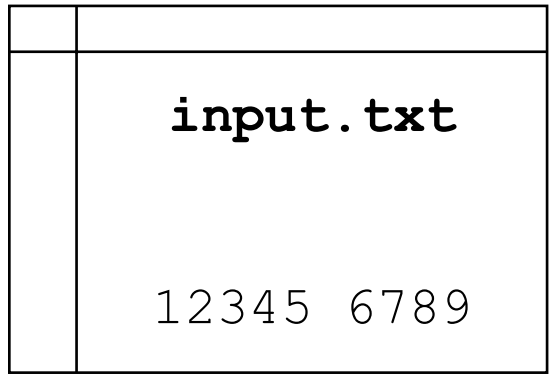
```
123456789
```


FileRead - Apache NetBeans IDE 11.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
312.4/501.0MB
Projects
Files
Services
Code4_1000074079
Source Packages
code4_1000074079
Code4_1000074079.java
Test Packages
Libraries
Test Libraries
Code5_1000074079
Code6_1000074079
Code7_1000074079
compareToDemo
Demos
FileRead
Source Packages
fileread
FileRead.java
Test Packages
Libraries
Test Libraries
FinalDemo
HelloWorld
Homework1
Homework2
Homework3
Homework4
Homework5
Homework6
Homework7
Homework8
ICQ2
ICQ3
ICQ3E
ICQ4
ICQ4-quiz
ICQ5
ICQ6
ICQ6A
Source Packages
icq6a
ICQ6A.java
FileRead.java
Source
History
* Donna French 1000074079
*/
package fileread;
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;
public class FileRead
{
public static void main(String[] args) throws FileNotFoundExcept
{
Scanner in = new Scanner(System.in);
System.out.print("Enter a file name ");
String FileName = in.next();
File fileIn = new File(FileName);
Scanner inFileRead = new Scanner(fileIn);
String FileLine = inFileRead.nextLine();
System.out.println(FileLine);
21:28
INS

File Handling

The normal whitespace rules apply

```
int FileLine = inFileRead.nextInt();
```

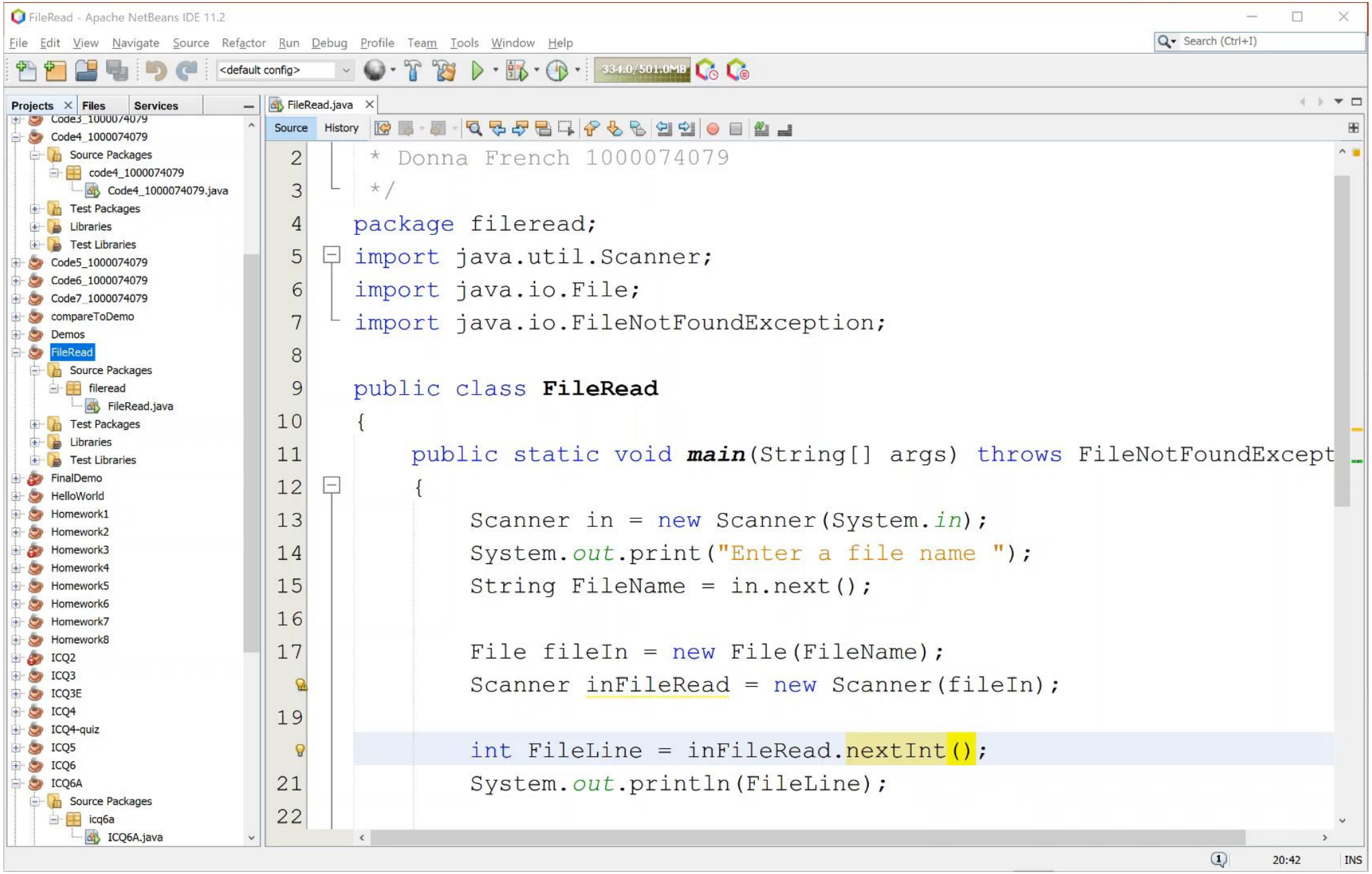


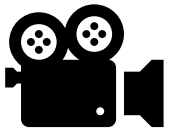
Then

```
System.out.println(FileLine);
```

will print the number up to the whitespace from the file

```
12345
```


FileRead - Apache NetBeans IDE 11.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
334.0/501.0MB
Projects Files Services
Code3_1000074079
Code4_1000074079
Source Packages
code4_1000074079
Code4_1000074079.java
Test Packages
Libraries
Test Libraries
Code5_1000074079
Code6_1000074079
Code7_1000074079
compareToDemo
Demos
FileRead
Source Packages
fileread
FileRead.java
Test Packages
Libraries
Test Libraries
FinalDemo
HelloWorld
Homework1
Homework2
Homework3
Homework4
Homework5
Homework6
Homework7
Homework8
ICQ2
ICQ3
ICQ3E
ICQ4
ICQ4-quiz
ICQ5
ICQ6
ICQ6A
Source Packages
icq6a
ICQ6A.java
FileRead.java
Source History
2 * Donna French 1000074079
3 */
4 package fileread;
5 import java.util.Scanner;
6 import java.io.File;
7 import java.io.FileNotFoundException;
8
9 public class FileRead
10 {
11 public static void main(String[] args) throws FileNotFoundExcept
12 {
13 Scanner in = new Scanner(System.in);
14 System.out.print("Enter a file name ");
15 String FileName = in.next();
16
17 File fileIn = new File(FileName);
Scanner inFileRead = new Scanner(fileIn);
19
int FileLine = inFileRead.nextInt();
21 System.out.println(FileLine);
22
20:42
INS



FileRead - Apache NetBeans IDE 11.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

<default config>

369.7/572.0MB

Projects Files Services

- Libraries
- compareToDemo
- Demos
- FileRead
- FinalDemo
- HelloWorld
- Homework1
- Homework2
- Homework3
- Homework4
- Homework5
- Homework6
- Homework7
- Homework8
- Homework9
- ICQ2
- ICQ3
- ICQ3E
- ICQ4
- ICQ4-quiz
- ICQ5
- ICQ6
- ICQ6A
- ICQ8
- ifSubstringDemo
- IncDecDemo
- MathDemo
- Method
- ModDemo
- MultiTable
- nestedforDemo
- NumberGuess
- Palindrome
- parseInt
- ReadSingleChar
- SplitDemo
- StringDemo
- trycatchDemo
  - Source Packages
    - trycatchdemo
      - TrycatchDemo.java

FileRead.java | Homework9.java | Code7_1000074079.java | TrycatchDemo.java

Source History

```
11      public static void main(String[] args) throws FileNotFoundExcept
12      {
13          Scanner in = new Scanner(System.in);
14          System.out.print("Enter a file name ");
15          String FileName = in.next();
16
17          File fileIn = new File(FileName);
18          Scanner inFileRead = new Scanner(fileIn);
19
20
21
22          inFileRead.close();
23
24
25      }
26  }
27
28
29
30
31
```

19:1 INS

```
Scanner in = new Scanner(System.in);
System.out.print("Enter a file name ");
String FileName = in.next();

File fileIn = new File(FileName);
Scanner inFileRead = new Scanner(fileIn);

String FileLine = "";

while (inFileRead.hasNextLine())
{
     FileLine = inFileRead.nextLine();
     System.out.println(FileLine);
}

inFileRead.close();
```

File Handling – Reading - Summary

2 new imports

```
import java.io.File;
import java.io.FileNotFoundException;
```

Add exception handling to `main` statement

```
public static void main(String[] args) throws FileNotFoundException
```

Create a file handle using a variable or a hardcoded file name

```
File fileIn = new File(FileName);
File fileIn = new File("input.txt");
File fileIn = new File("C:\\temp\\input.txt");
```

Create a `Scanner` that uses that file handle

```
Scanner inFileRead = new Scanner(fileIn);
```

File Handling

To write to a file, construct a `PrintWriter` object with the output file name

```
PrintWriter out = new PrintWriter("out.txt");
```

This opens the file `out.txt` for writing.

If the output file exists, it is emptied before the new data is written into it.

If the file does not exist, then an empty file is created.

File Handling

```
PrintWriter out = new PrintWriter("out.txt");
```

An import file is needed to use `PrintWriter`

```
import java.io.PrintWriter;
```

File Handling

```
PrintWriter out = new PrintWriter("out.txt");
```

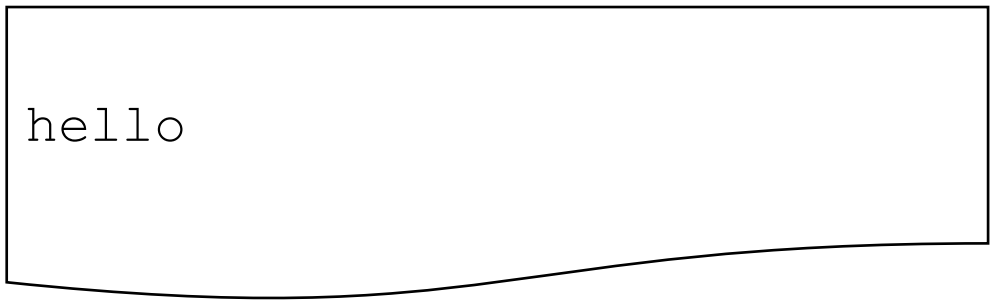
We can use the print methods that we are already familiar with to write to files.

```
out.println()
out.print()
```

File Handling

```
PrintWriter out = new PrintWriter("out1.txt");
out.println("hello");
out.close();
```

Writes `hello` to `out1.txt`

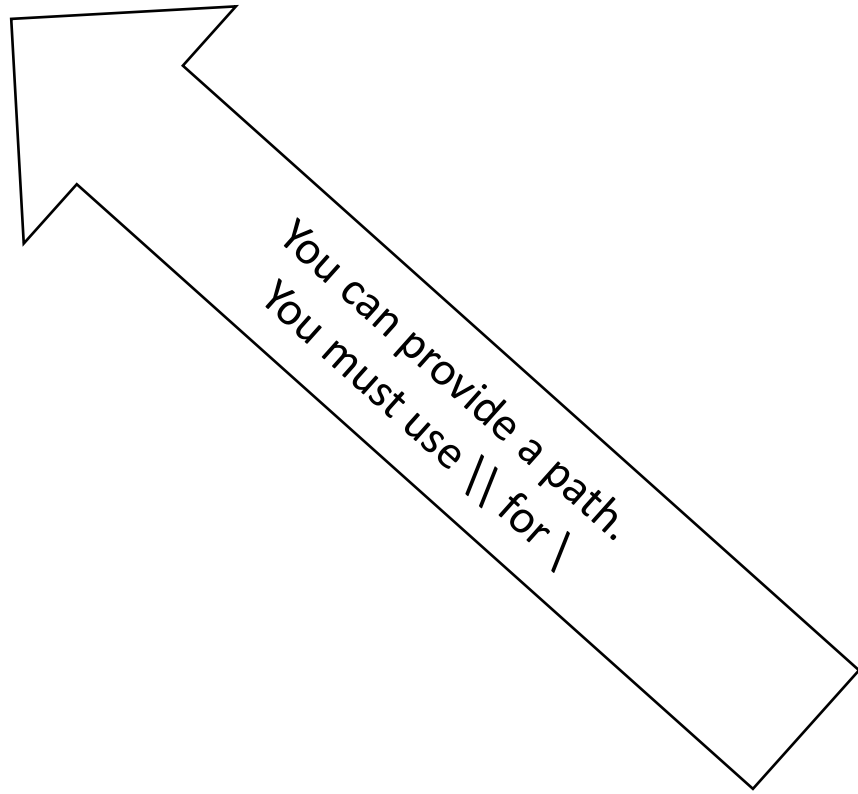


`out1.txt`

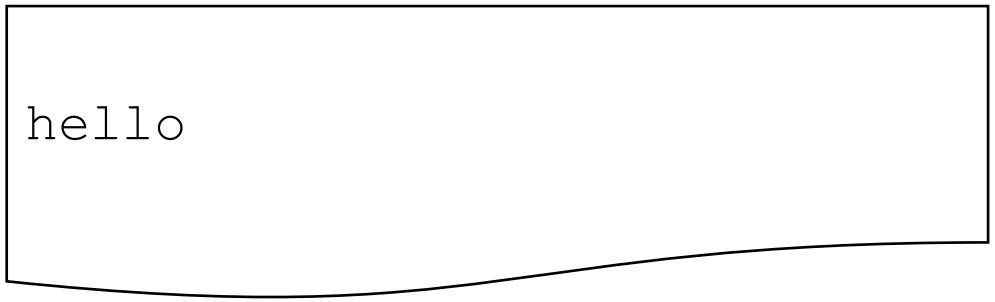
The file will be created in the same directory as your Java directory unless you specify a path.

File Handling

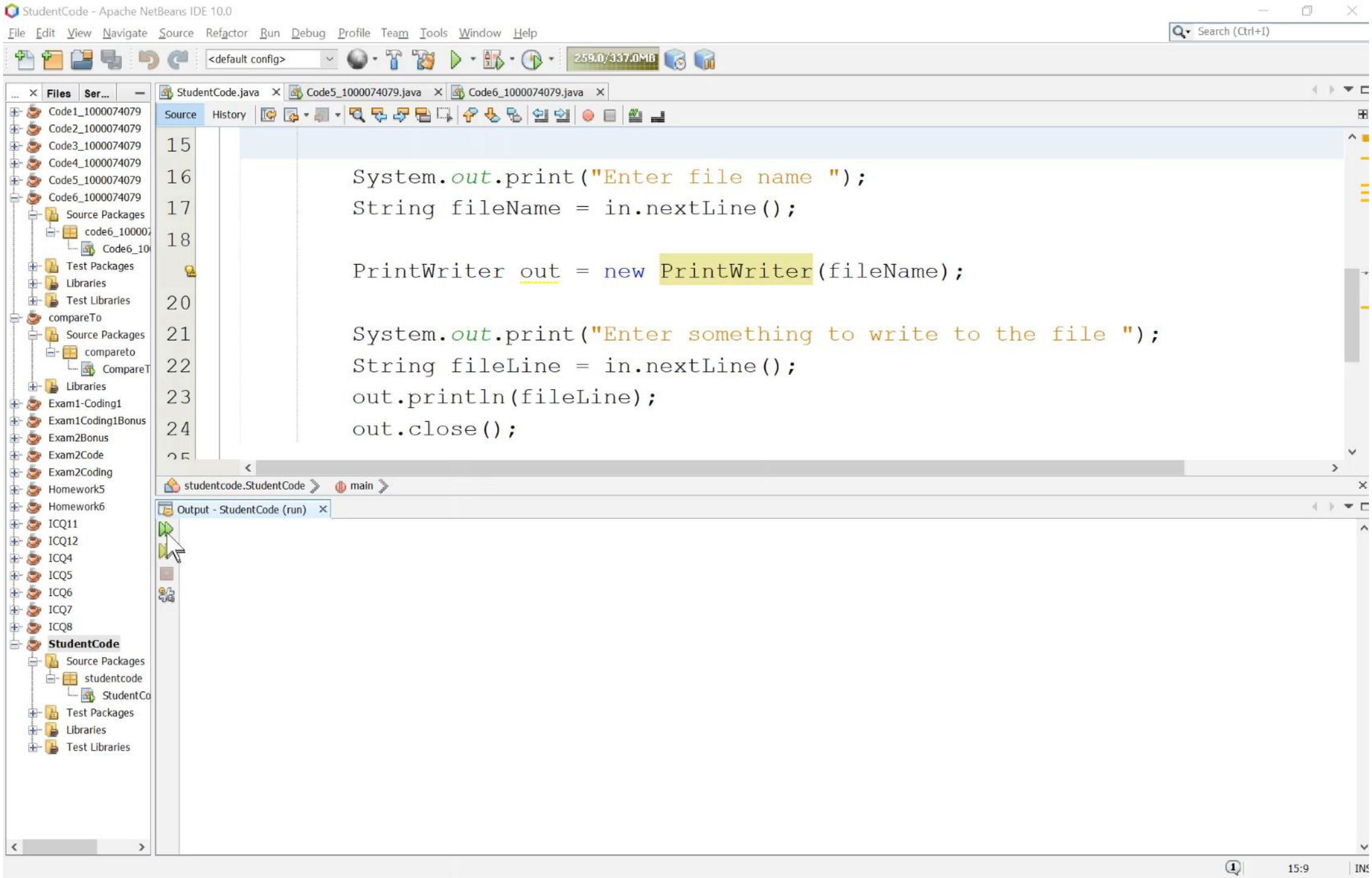
```
PrintWriter out = new PrintWriter(" C:\\Desktop\\out1.txt");
out.println("hello");
out.close();
```



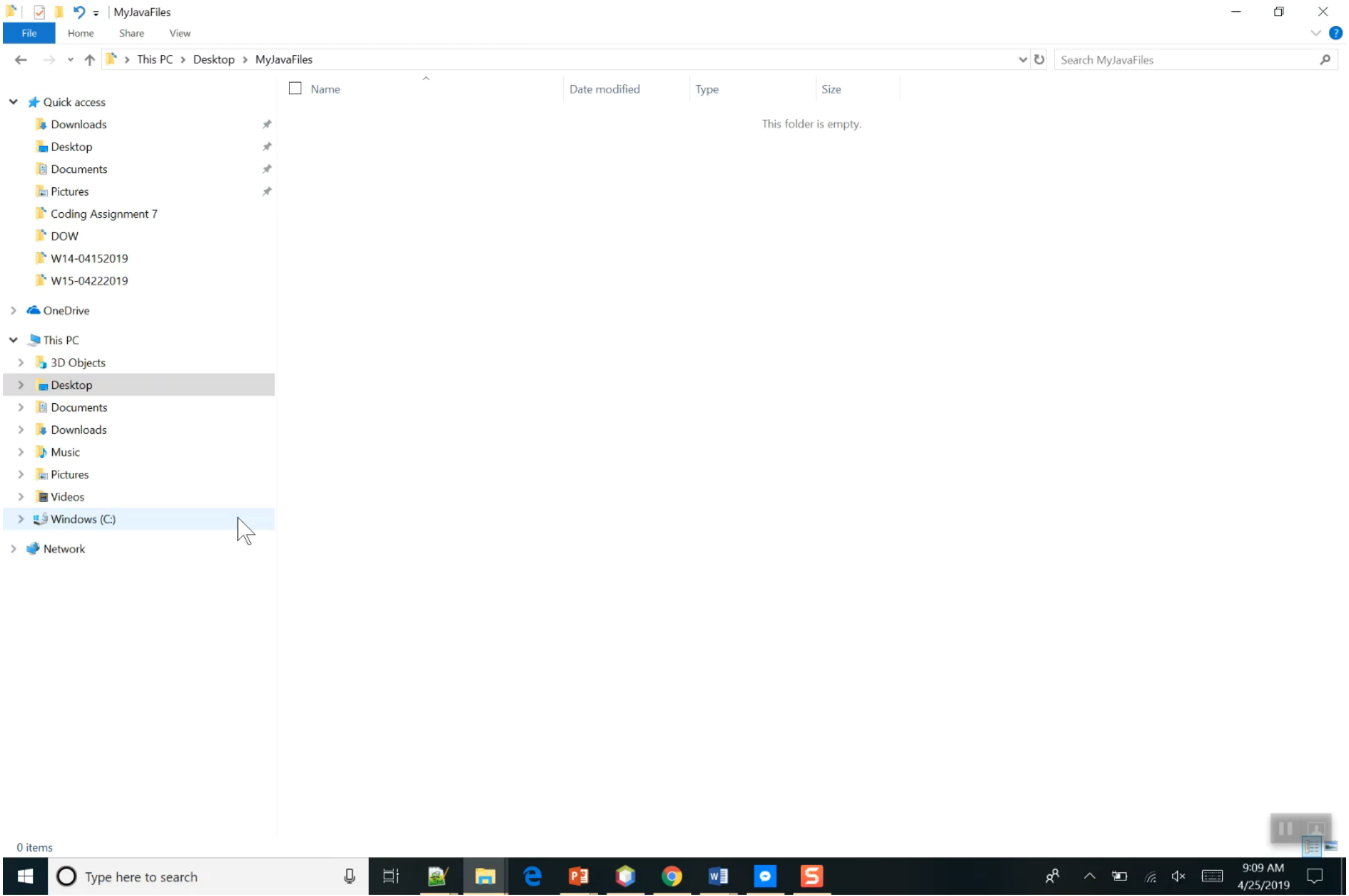
Writes `hello` to `C:\Desktop\out1.txt`



`C:\Desktop\out1.txt`


StudentCode - Apache NetBeans IDE 10.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
259.0/337.0MB
Files Ser...
Code1_1000074079
Code2_1000074079
Code3_1000074079
Code4_1000074079
Code5_1000074079
Code6_1000074079
Source Packages
code6_100007
Code6_10
Test Packages
Libraries
Test Libraries
compareTo
Source Packages
compareto
CompareT
Libraries
Exam1-Coding1
Exam1Coding1Bonus
Exam2Bonus
Exam2Code
Exam2Coding
Homework5
Homework6
ICQ11
ICQ12
ICQ4
ICQ5
ICQ6
ICQ7
ICQ8
StudentCode
Source Packages
studentcode
StudentCo
Test Packages
Libraries
Test Libraries
StudentCode.java
Code5_1000074079.java
Code6_1000074079.java
Source
History
15
16 System.out.print("Enter file name ");
17 String fileName = in.nextLine();
18
PrintWriter out = new PrintWriter(fileName);
20
21 System.out.print("Enter something to write to the file ");
22 String fileLine = in.nextLine();
23 out.println(fileLine);
24 out.close();
studentcode.StudentCode
main
Output - StudentCode (run)
15:9
INS

```
System.out.print("Enter file name ");
String fileName = in.nextLine();

PrintWriter out = new PrintWriter(fileName);

System.out.print("Enter something to write to the file ");
String fileLine = in.nextLine();

out.println(fileLine);

out.close();
```


MyJavaFiles
File
Home
Share
View
This PC
Desktop
MyJavaFiles
Search MyJavaFiles
Quick access
Downloads
Desktop
Documents
Pictures
Coding Assignment 7
DOW
W14-04152019
W15-04222019
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
Network
Name
Date modified
Type
Size
This folder is empty.
0 items
Type here to search
9:09 AM
4/25/2019

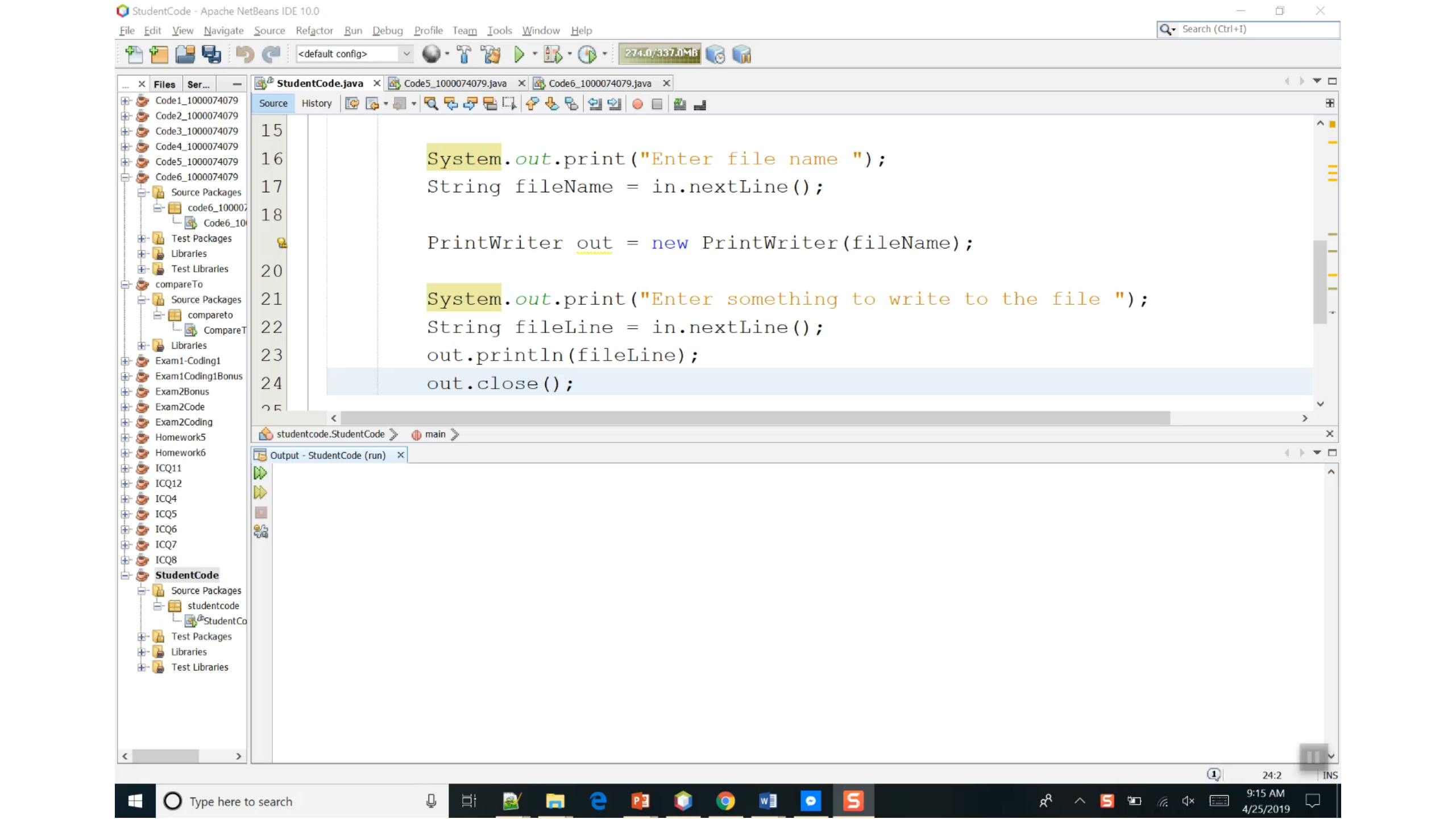
File Handling

```
PrintWriter out = new PrintWriter("out.txt");
```

Just as with opening files for reading, we need to close files that we opened for writing.

```
out.close();
```

Not closing the file can result in data loss.


StudentCode - Apache NetBeans IDE 10.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
274.0/337.0MB
Files Ser...
Code1_1000074079
Code2_1000074079
Code3_1000074079
Code4_1000074079
Code5_1000074079
Code6_1000074079
Source Packages
code6_100007
Code6_10
Test Packages
Libraries
Test Libraries
compareTo
Source Packages
compareto
CompareT
Libraries
Exam1-Coding1
Exam1Coding1Bonus
Exam2Bonus
Exam2Code
Exam2Coding
Homework5
Homework6
ICQ11
ICQ12
ICQ4
ICQ5
ICQ6
ICQ7
ICQ8
StudentCode
Source Packages
studentcode
StudentCo
Test Packages
Libraries
Test Libraries
StudentCode.java
Code5_1000074079.java
Code6_1000074079.java
Source
History
15
16 System.out.print("Enter file name ");
17 String fileName = in.nextLine();
18
PrintWriter out = new PrintWriter(fileName);
20
21 System.out.print("Enter something to write to the file ");
22 String fileLine = in.nextLine();
23 out.println(fileLine);
24 out.close();
studentcode.StudentCode
main
Output - StudentCode (run)
24:2
INS
Type here to search
9:15 AM
4/25/2019

File Handling

When using `PrintWriter` to open a file for writing, if the file does not exist, `PrintWriter` will throw an exception and we have to add code to handle that situation

```
PrintWriter out = new PrintWriter("out.txt");
```

For now, simply add this to `main` method

```
public static void main(String[] args) throws FileNotFoundException
```

File Handling

Sometimes, our program will encounter a situation that makes it difficult or meaningless for our program to continue to run. In these special cases, the exit command can be used to stop the program.

```
System.exit(0);
```

This command will cause the program to immediately terminate.

Useful for when a program relies on files to open for processing but the files fail to open.

Write a program to open a file for reading and another file for writing. Assume the file opened for reading contains a question. Read the question from the file and write it out to the screen. Prompt the user for the answer and write the answer to the output file. Add exception handling to exit the program if the question file does not exist.

File Handling

Write a program to open a file for reading and another file for writing.

This tell us that we need a complete program rather than just a method.

We will be reading and writing with files and prompting for input.

```
import java.util.Scanner;
import java.io.PrintWriter;
import java.io.File;
```

And we will need to add

```
        throws FileNotFoundException
```

```
package qanda;

import java.util.Scanner;
import java.io.PrintWriter;
import java.io.File;

public class QandA
{
    public static void main(String[] args) throws FileNotFoundException
    {
    }
}
```

Write a program to open a file for reading and another file for writing. Assume the file opened for reading contains a question. Read the question from the file and write it out to the screen. Prompt the user for the answer and write the answer to the output file. Add exception handling to exit the program if the question file does not exist.

Assume the file opened for reading contains a question.

So we need to ask for a file name (question did not give us one so we need to ask for it – don't assume a file name).

```
Scanner in = new Scanner(System.in);
System.out.print("Enter the question file name ");
String questionFileName = in.nextLine();
```

We need to use that file name to open a file to read it.

```
File FH = new File(questionFileName);
Scanner inFile = new Scanner(FH);
```

```
package qanda;

import java.util.Scanner;
import java.io.PrintWriter;
import java.io.File;

public class QandA
{
    public static void main(String[] args) throws FileNotFoundException
    {
        Scanner in = new Scanner(System.in);

        System.out.print("Enter the question file name ");
        String questionFileName = in.nextLine();

        File FH = new File(questionFileName);
        Scanner inFile = new Scanner(FH);
    }
}
```

Write a program to open a file for reading and another file for writing. Assume the file opened for reading contains a question. Read the question from the file and write it out to the screen. Prompt the user for the answer and write the answer to the output file. Add exception handling to exit the program if the question file does not exist.

Read the question from the file and write it out to the screen.

We need to read the question from the file.

```
String question = inFile.nextLine();
```

We need to write it out to the screen.

```
System.out.printf("\nThe question is \"%s\"", question);
```

```
package qanda;

import java.util.Scanner;
import java.io.PrintWriter;
import java.io.File;

public class QandA
{
    public static void main(String[] args) throws FileNotFoundException
    {
        Scanner in = new Scanner(System.in);

        System.out.print("Enter the question file name ");
        String questionFileName = in.nextLine();

        File FH = new File(questionFileName);
        Scanner inFile = new Scanner(FH);

        String question = inFile.nextLine();
        inFile.close();
        System.out.printf("\nThe question is \"%s\"", question);
    }
}
```

Write a program to open a file for reading and another file for writing. Assume the file opened for reading contains a question. Read the question from the file and write it out to the screen. Prompt the user for the answer and write the answer to the output file. Add exception handling to exit the program if the question file does not exist.

Prompt the user for the answer and write the answer to the output file.

We need to prompt for and read the answer.

```
System.out.print("\n\nWhat is your answer?  ");
String answer = in.nextLine();
```

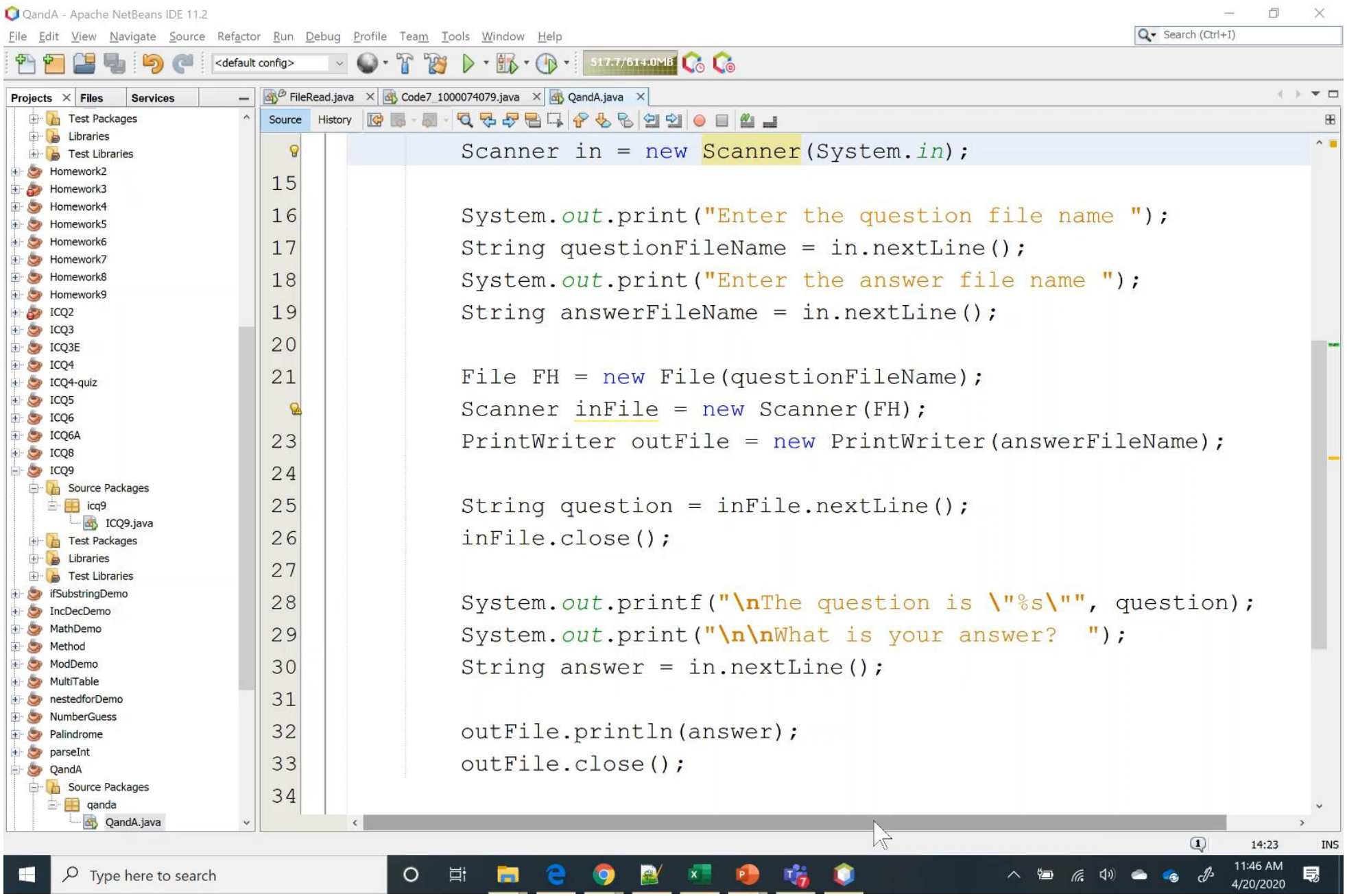
We don't have a name for the output file so we need to ask for it and open it for writing.

```
System.out.print("Enter the answer file name ");
String answerFileName = in.nextLine();
PrintWriter outFile = new PrintWriter(answerFileName);
```

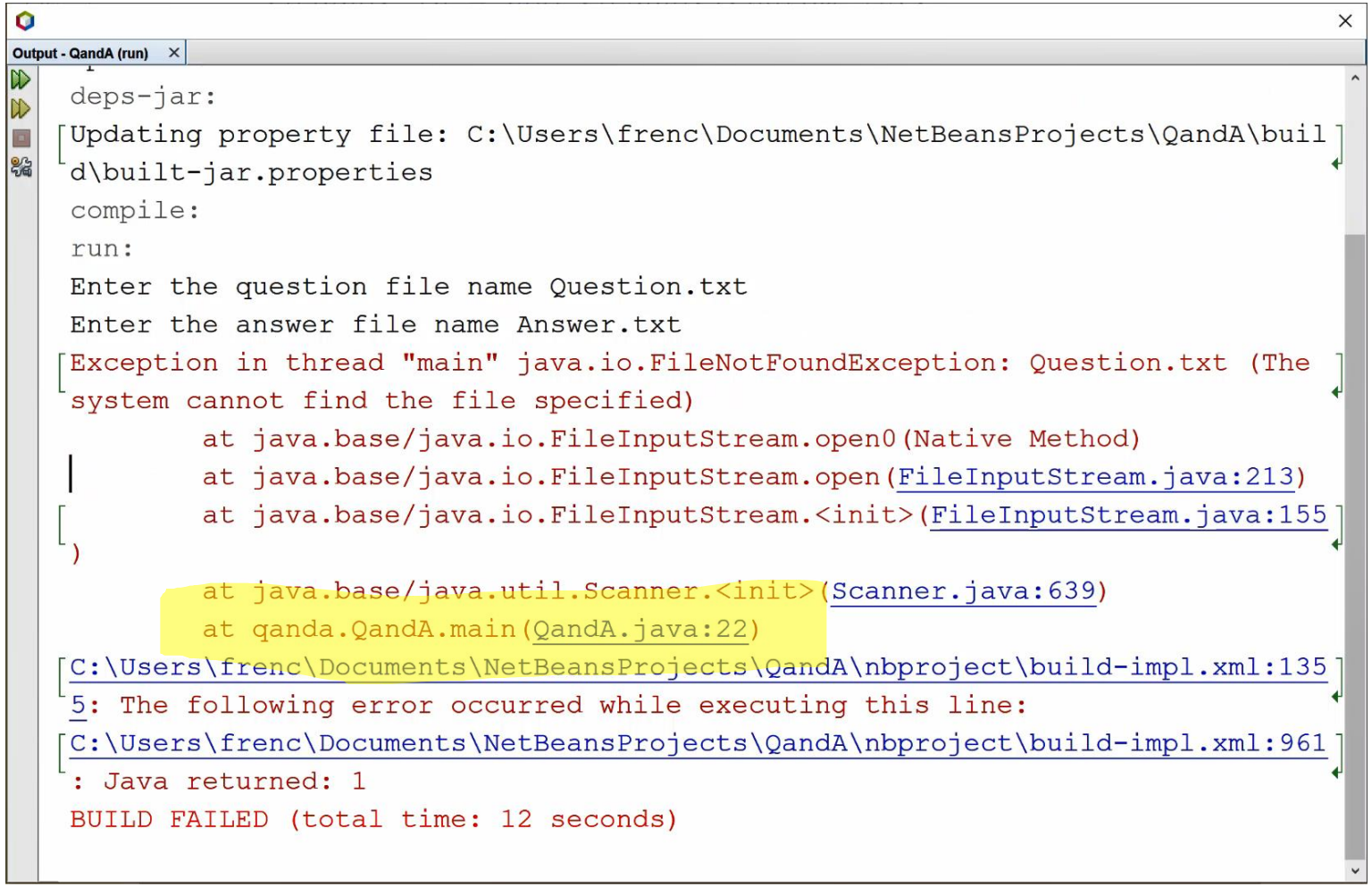
We need to write it to the output file.

```
outFile.println(answer);
```

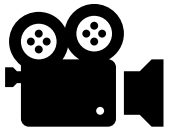
```
    public static void main(String[] args) throws FileNotFoundException
    {
        Scanner in = new Scanner(System.in);

        System.out.print("Enter the question file name ");
        String questionFileName = in.nextLine();

        File FH = new File(questionFileName);
        Scanner inFile = new Scanner(FH);

        String question = inFile.nextLine();
        inFile.close();
        System.out.printf("\nThe question is \"%s\"", question);
        System.out.print("\n\nWhat is your answer?  ");
        String answer = in.nextLine();
        System.out.print("Enter the answer file name ");
        String answerFileName = in.nextLine();
        PrintWriter outFile = new PrintWriter(answerFileName);
        outFile.println(answer);
        outFile.close();
    }
}
```


QandA - Apache NetBeans IDE 11.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Projects Files Services
FileRead.java
Code7_1000074079.java
QandA.java
Source History
Test Packages
Libraries
Test Libraries
Homework2
Homework3
Homework4
Homework5
Homework6
Homework7
Homework8
Homework9
ICQ2
ICQ3
ICQ3E
ICQ4
ICQ4-quiz
ICQ5
ICQ6
ICQ6A
ICQ8
ICQ9
Source Packages
icq9
ICQ9.java
Test Packages
Libraries
Test Libraries
ifSubstringDemo
IncDecDemo
MathDemo
Method
ModDemo
MultiTable
nestedforDemo
NumberGuess
Palindrome
parseInt
QandA
Source Packages
qanda
QandA.java
Scanner in = new Scanner(System.in);
System.out.print("Enter the question file name ");
String questionFileName = in.nextLine();
System.out.print("Enter the answer file name ");
String answerFileName = in.nextLine();
File FH = new File(questionFileName);
Scanner inFile = new Scanner(FH);
PrintWriter outFile = new PrintWriter(answerFileName);
String question = inFile.nextLine();
inFile.close();
System.out.printf("\nThe question is \"%s\"", question);
System.out.print("\n\nWhat is your answer? ");
String answer = in.nextLine();
outFile.println(answer);
outFile.close();
14:23
INS
Type here to search
11:46 AM
4/20/2020


Output - QandA (run)
[ant -f C:\\Users\\frenc\\Documents\\NetBeansProjects\\QandA -Dnb.internal.ac
tion.name=run run




```
20
21          File FH = new File(questionFileName);
            Scanner inFile = new Scanner(FH);
23          PrintWriter outFile = new PrintWriter(answerFileName);
24
```



```
14          Scanner in = new Scanner(System.in);
15
16          System.out.print("Enter the question file name ");
17          String questionFileName = in.nextLine();
18          System.out.print("Enter the answer file name ");
19          String answerFileName = in.nextLine();
20
21          File FH = new File(questionFileName);
            Scanner inFile = new Scanner(FH);
23          PrintWriter outFile = new PrintWriter(answerFileName);
24
25          String question = inFile.nextLine();
26          inFile.close();
27
28          System.out.printf("\nThe question is \"%s\"", question);
29          System.out.print("\n\nWhat is your answer?  ");
30          String answer = in.nextLine();
31
32          outFile.println(answer);
33          outFile.close();
34
```

Problem : Read from a file to initialize a 2D array.

Step 1 – a file name has not been provided so we need to ask for one.

```
Scanner in = new Scanner(System.in);
System.out.print("Enter a file name ");
String Filename = in.nextLine();
```

Step 2 - Now we need to open the file

```
File FH = new File(Filename);
Scanner inFile = new Scanner(FH);
```

Step 3 - Now we need to read the file

```
String FileLine = inFile.nextLine();
```

Problem : Read from a file to initialize a 2D array.

Step 4 – we have the line from the file in a variable

`FileLine` contains "`ABCDEFGHIJKLMNOPQRSTUVWXY`"

Step 5 – We need to initialize the 2D array with the contents of `FileLine`

Create the 2D array – we were not told anything about the size

```
char My2D[][] = new char[5][5];
```

Problem : Read from a file to initialize a 2D array.

Step 5 – We need to initialize the 2D array with the contents of `FileLine`

```
for (int i = 0; i < My2D.length; i++)
{
    for (int j = 0; j < My2D[0].length; j++)
    {

    }
}
```

Problem : Read from a file to initialize a 2D array.

Step 5 – We need to initialize the 2D array with the contents of `FileLine`

```
for (int i = 0; i < My2D.length; i++)
{
    for (int j = 0; j < My2D[0].length; j++)
    {
        My2D[i][j] = ?
    }
}
```

"ABCDEFGHIJKLMNOPQRSTUVWXY"

Problem : Read from a file to initialize a 2D array.

Step 5 – We need to initialize the 2D array with the contents of `FileLine`

```
for (int i = 0; i < My2D.length; i++)
{
    for (int j = 0; j < My2D[0].length; j++)
    {
        My2D[i][j] = FileLine.charAt(?);
    }
}
```

"ABCDEFGHIJKLMNOPQRSTUVWXY"

Problem : Read from a file to initialize a 2D array.

Step 5 – We need to initialize the 2D array with the contents of `FileLine`

```
int counter = 0;
for (int i = 0; i < My2D.length; i++)
{
    for (int j = 0; j < My2D[0].length; j++)
    {
        My2D[i][j] = FileLine.charAt(counter++);
    }
}
```

"ABCDEFGHIJKLMNOPQRSTUVWXY"

Things to Ponder….

Problem : Read from a file to initialize a 2D array. The first line of the file contains the dimensions of the 2D array.

Example file

```
4x10
AMDYEWQITRUIEDCNQWURPOQSNCGFASZXMNVUYEOP
```

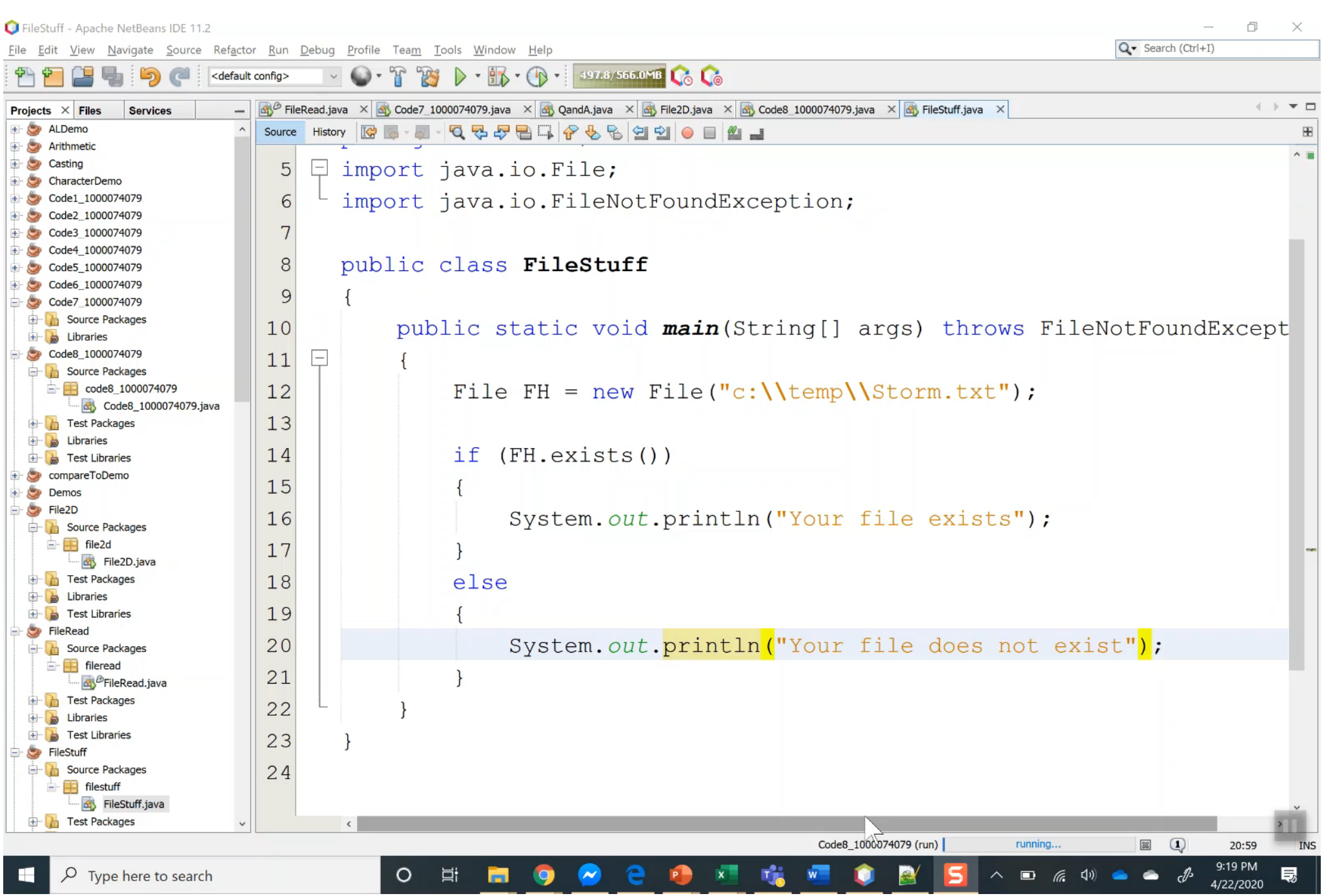
You would need to read the first line and parse it into 4 and 10 and use those values to create the 2D array. You would then read the next line of the file and insert those values into the 2D array.

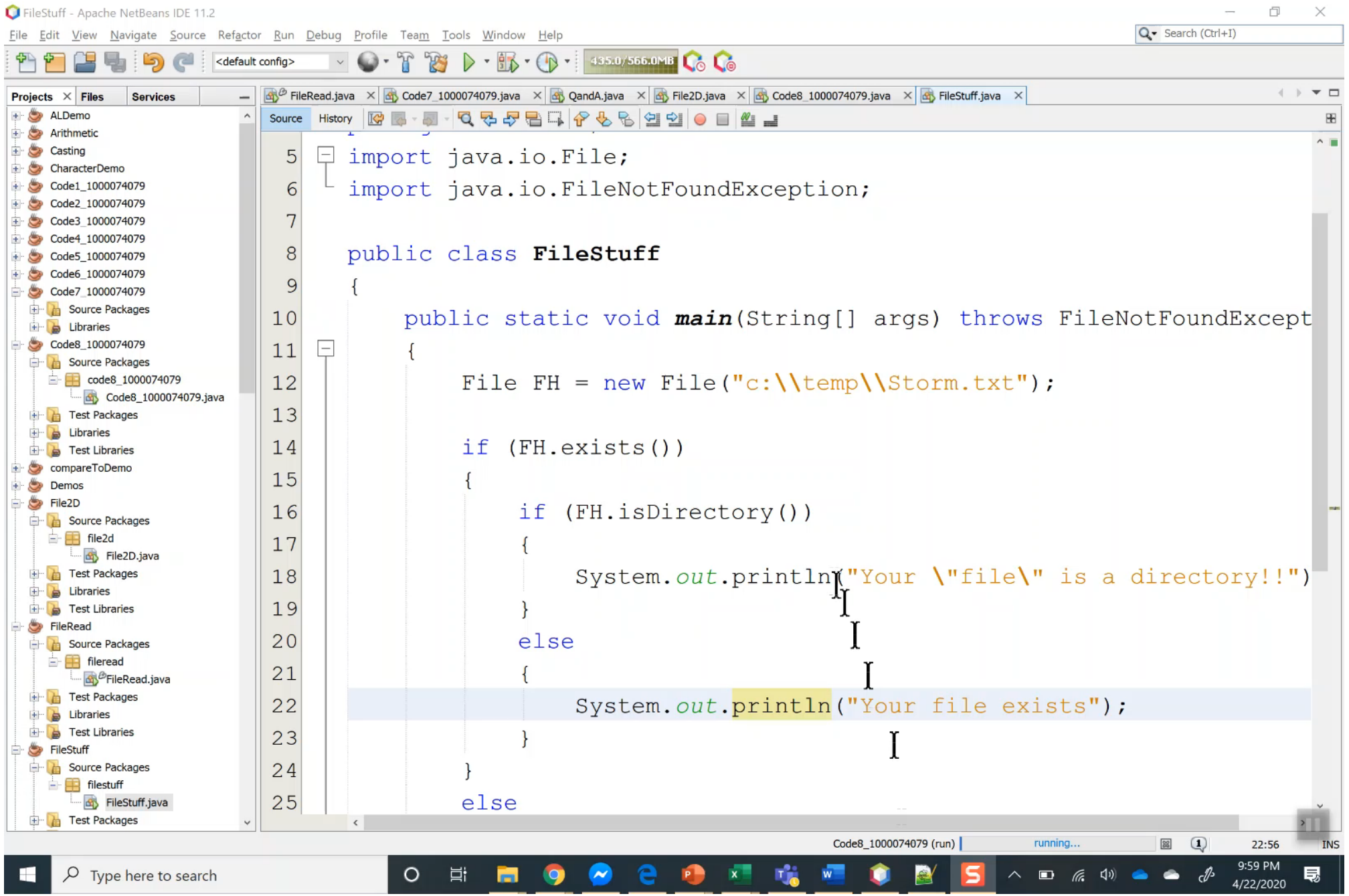
More things to Ponder…

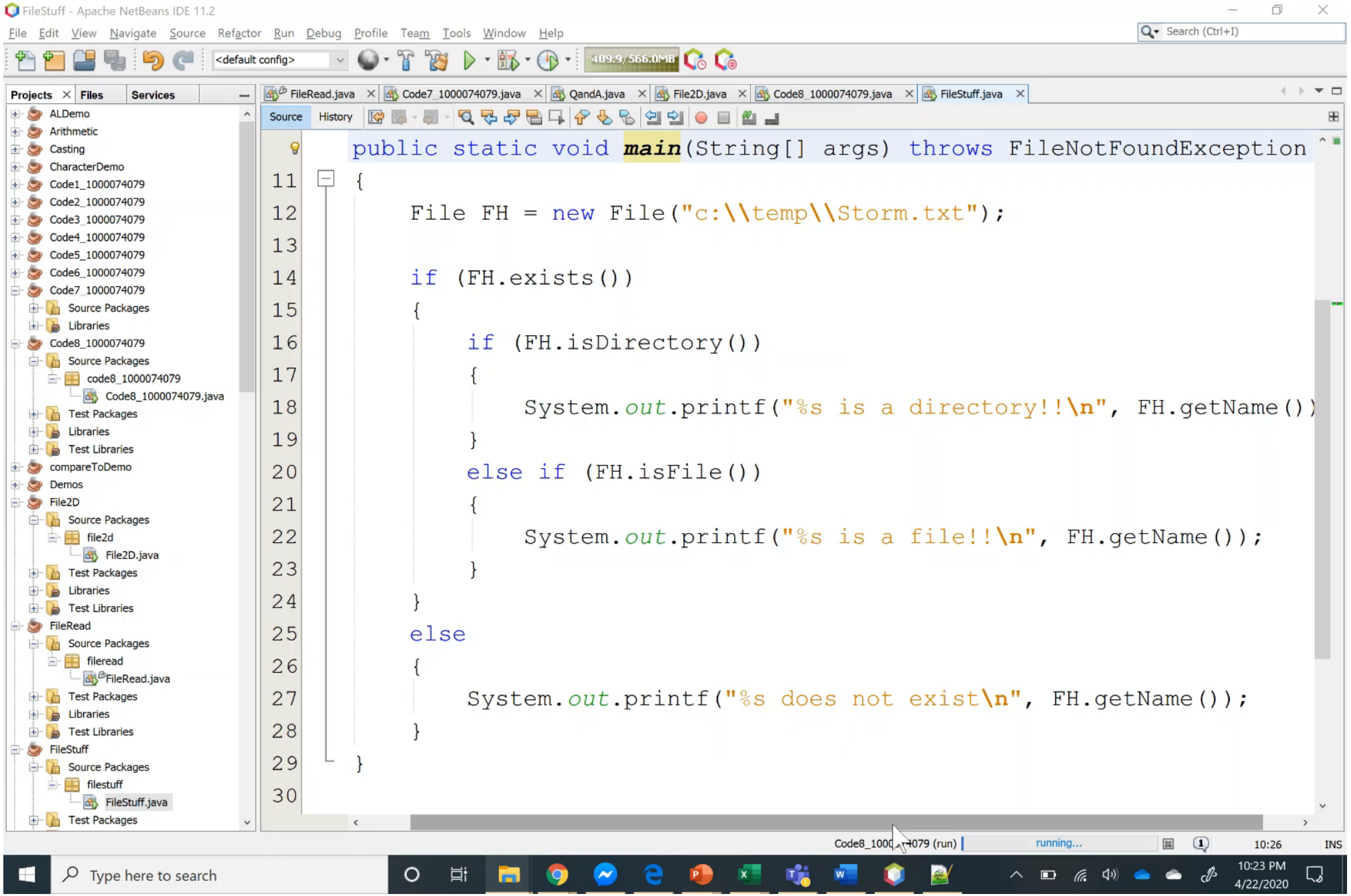
How would you write the 2D array back out to a file?

How do you write a 2D array to the screen?

```
A    B    C    D    E
F    G    H    I    J
K    L    M    N    O
P    Q    R    S    T
U    V    W    X    Y
```


FileStuff - Apache NetBeans IDE 11.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
497.8/566.0MB
Projects Files Services
FileRead.java Code7_1000074079.java QandA.java File2D.java Code8_1000074079.java FileStuff.java
Source History
ALDemo
Arithmetic
Casting
CharacterDemo
Code1_1000074079
Code2_1000074079
Code3_1000074079
Code4_1000074079
Code5_1000074079
Code6_1000074079
Code7_1000074079
Source Packages
Libraries
Code8_1000074079
Source Packages
code8_1000074079
Code8_1000074079.java
Test Packages
Libraries
Test Libraries
compareToDemo
Demos
File2D
Source Packages
file2d
File2D.java
Test Packages
Libraries
Test Libraries
FileRead
Source Packages
fileread
FileRead.java
Test Packages
Libraries
Test Libraries
FileStuff
Source Packages
filestuff
FileStuff.java
Test Packages
5 import java.io.File;
6 import java.io.FileNotFoundException;
7
8 public class FileStuff
9 {
10 public static void main(String[] args) throws FileNotFoundExcept
11 {
12 File FH = new File("c:\\temp\\Storm.txt");
13
14 if (FH.exists())
15 {
16 System.out.println("Your file exists");
17 }
18 else
19 {
20 System.out.println("Your file does not exist");
21 }
22 }
23 }
24
Code8_1000074079 (run) running...
20:59
INS
Type here to search
9:19 PM
4/22/2020


FileStuff - Apache NetBeans IDE 11.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
435.0/566.0MB
Projects Files Services
FileRead.java Code7_1000074079.java QandA.java File2D.java Code8_1000074079.java FileStuff.java
Source History
ALDemo
Arithmetic
Casting
CharacterDemo
Code1_1000074079
Code2_1000074079
Code3_1000074079
Code4_1000074079
Code5_1000074079
Code6_1000074079
Code7_1000074079
Source Packages
Libraries
Code8_1000074079
Source Packages
code8_1000074079
Code8_1000074079.java
Test Packages
Libraries
Test Libraries
compareToDemo
Demos
File2D
Source Packages
file2d
File2D.java
Test Packages
Libraries
Test Libraries
FileRead
Source Packages
fileread
FileRead.java
Test Packages
Libraries
Test Libraries
FileStuff
Source Packages
filestuff
FileStuff.java
Test Packages
5 import java.io.File;
6 import java.io.FileNotFoundException;
7
8 public class FileStuff
9 {
10 public static void main(String[] args) throws FileNotFoundExcept
11 {
12 File FH = new File("c:\\temp\\Storm.txt");
13
14 if (FH.exists())
15 {
16 if (FH.isDirectory())
17 {
18 System.out.println("Your \"file\" is a directory!!")
19 }
20 else
21 {
22 System.out.println("Your file exists");
23 }
24 }
25 else
Code8_1000074079 (run) running...
22:56
INS
Type here to search
9:59 PM
4/22/2020


FileStuff - Apache NetBeans IDE 11.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
409.9/566.0MB
Projects Files Services
ALDemo
Arithmetic
Casting
CharacterDemo
Code1_1000074079
Code2_1000074079
Code3_1000074079
Code4_1000074079
Code5_1000074079
Code6_1000074079
Code7_1000074079
Source Packages
Libraries
Code8_1000074079
Source Packages
code8_1000074079
Code8_1000074079.java
Test Packages
Libraries
Test Libraries
compareToDemo
Demos
File2D
Source Packages
file2d
File2D.java
Test Packages
Libraries
Test Libraries
FileRead
Source Packages
fileread
FileRead.java
Test Packages
Libraries
Test Libraries
FileStuff
Source Packages
filestuff
FileStuff.java
Test Packages
FileRead.java
Code7_1000074079.java
QandA.java
File2D.java
Code8_1000074079.java
FileStuff.java
Source
History
public static void main(String[] args) throws FileNotFoundException
{
File FH = new File("c:\\temp\\Storm.txt");
if (FH.exists())
{
if (FH.isDirectory())
{
System.out.printf("%s is a directory!!\n", FH.getName());
}
else if (FH.isFile())
{
System.out.printf("%s is a file!!\n", FH.getName());
}
}
else
{
System.out.printf("%s does not exist\n", FH.getName());
}
}
Code8_1000074079 (run)
running...
10:26
INS
Type here to search
10:23 PM
4/22/2020

Coding Exercise

We need to transmit our Top Secret Spy's dossier via an encoded text message. The program should be able to encode and decode text files via this menu using a simple Caesar cipher using a left shift of 13.

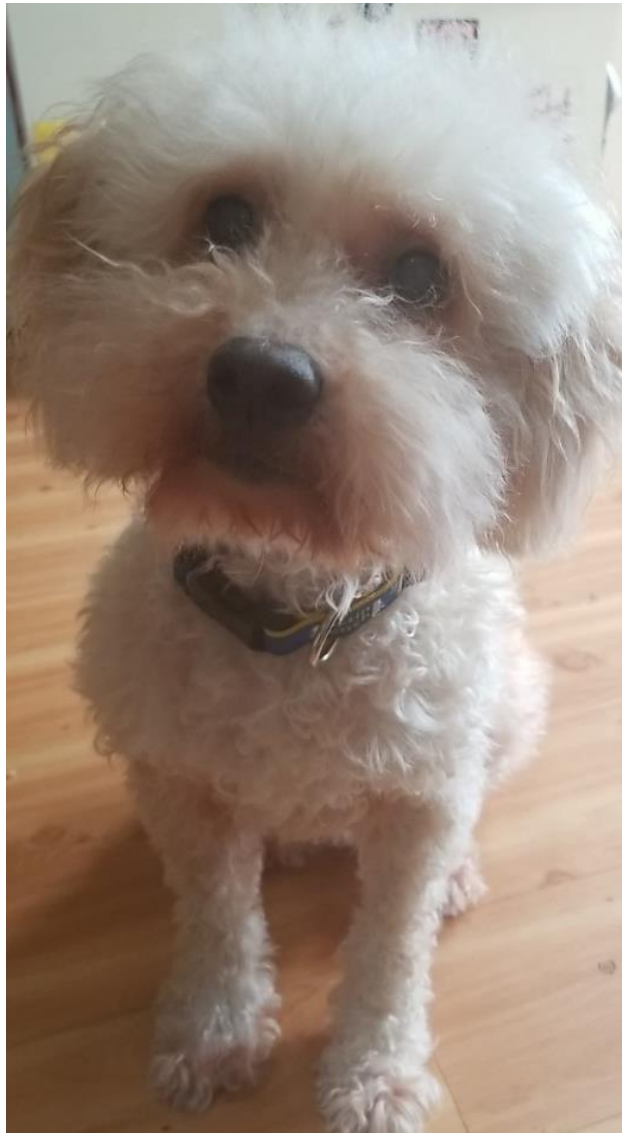
```
1. Encode file
2. Decode file
3. View file contents
4. Exit
```

A file was not named specified in the coding request so let's make our own.

AgentAF.txt - Notepad

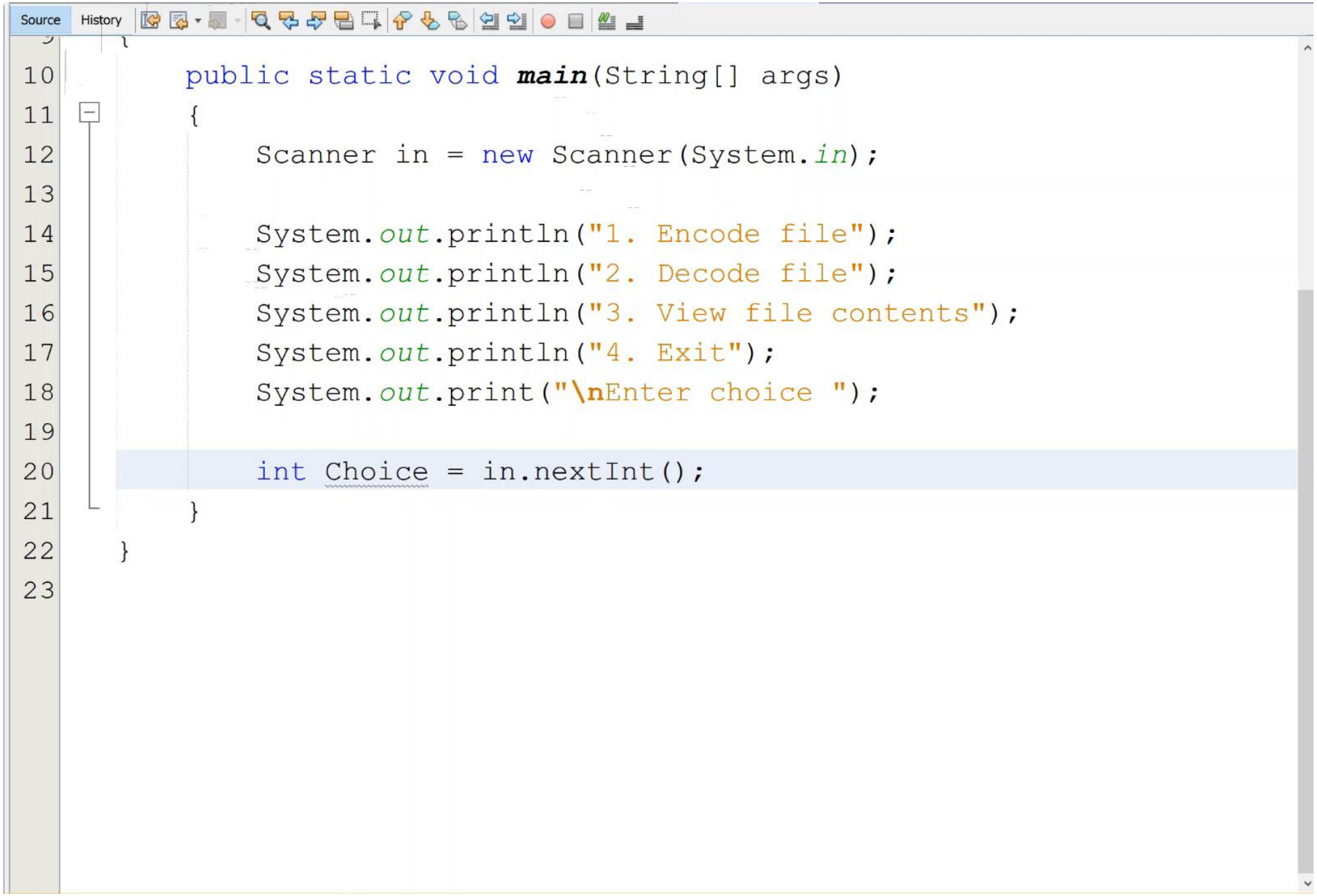
File Edit Format View Help

```
Top Secret Agent Dossier

Name : Austin French
Date of Birth : Unknown
Gender : M
Languages : Dog and limited Human
Eye Color : Brown
Hair Color : Cream
Breed : Poodle Mix
Cover : Family Fitness Trainer
Weakness : Blankets
Codename : Couch Wolf
```



We can create this menu first since it was provided.

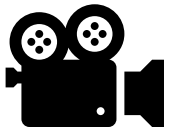
```
1. Encode file
2. Decode file
3. View file contents
4. Exit

Scanner in = new Scanner(System.in);

System.out.println("1. Encode file");
System.out.println("2. Decode file");
System.out.println("3. View file contents");
System.out.println("4. Exit");
System.out.print("\nEnter choice ");

int Choice = in.nextInt();
```


Source
History
{
public static void main(String[] args)
{
Scanner in = new Scanner(System.in);
System.out.println("1. Encode file");
System.out.println("2. Decode file");
System.out.println("3. View file contents");
System.out.println("4. Exit");
System.out.print("\nEnter choice ");
int Choice = in.nextInt();
}
}

This works OK if a valid choice is entered. We need to handle an invalid choice.

```
int Choice = in.nextInt();

do
{
  Choice = in.nextInt()
  if (Choice < 1 || Choice > 4)
    System.out.print("Invalid choice.  Please reenter ");
}
while (Choice < 1 || Choice > 4);
```

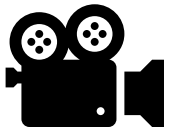
This will make sure we have a Choice value between 1 and 4 but what will happen if a non-integer value is entered?



```
    public static void main(String[] args)
    {
        Scanner in = new Scanner(System.in);
        int Choice = 0;

        System.out.println("1. Encode file");
        System.out.println("2. Decode file");
        System.out.println("3. View file contents");
        System.out.println("4. Exit");
        System.out.print("\nEnter choice ");

        do
        {
            Choice = in.nextInt();
            if (Choice < 1 || Choice > 4)
                System.out.print("Invalid choice.  Please reenter ")
        }
        while (Choice < 1 || Choice > 4);
    }
}
```

We need to add exception handling.

```
do
{
   try
   {
     Choice = in.nextInt()
   }
   catch(Exception e)
   {
     Choice = 0;
   }

   if (Choice < 1 || Choice > 4)
     System.out.print("Invalid choice.  Please reenter ");
}
while (Choice < 1 || Choice > 4);
```



Source History

```
21          do
22          {
23              try
24              {
25                  Choice = in.nextInt();
26              }
27              catch(Exception e)
28              {
29                  Choice = 0;
30              }
31
32              if (Choice < 1 || Choice > 4)
33                  System.out.print("Invalid choice.  Please reenter ")
34          }
35          while (Choice < 1 || Choice > 4);
36      }
37  }
38
```

So why did our `do-while` infinitely loop?

The "`if`" executed after the `catch` because we saw the "`Invalid choice.  Please reenter `". The `do-while` kept executing yet it never stopped at the `Choice = in.nextInt();` prompt again.

Why?

When `nextInt()` cannot consume the next value in `stdin` (when it's not an integer), it leaves in it `stdin`. When the `do-while` started over, `Choice = in.nextInt();` read `stdin` again and got the non-integer again and threw an exception again. This process of `nextInt()`, exception, `nextInt()`, exception formed the infinite loop.

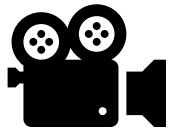
```
do
{
    try
    {
        Choice = in.nextInt()
    }
    catch(Exception e)
    {
        Choice = 0;
    }

    if (Choice < 1 || Choice > 4)
        System.out.print("Invalid choice.  Please reenter ");
}
while (Choice < 1 || Choice > 4);
```

When `nextInt()` cannot consume the next value in `stdin` (when it's not an integer), it leaves in it `stdin`. When the `do-while` started over, `Choice = in.nextInt();` read `stdin` again and got the non-integer again and threw an exception again. This process of `nextInt()`, exception, `nextInt()`, exception formed the infinite loop.

So how do we fix this?

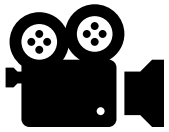
We need to clear that bad input out of `stdin`.

```
do
{
   try
   {
      Choice = in.nextInt()
   }
   catch(Exception e)
   {
      Choice = 0;
      in.next();
   }

   if (Choice < 1 || Choice > 4)
      System.out.print("Invalid choice.  Please reenter ");
}
while (Choice < 1 || Choice > 4);
```



Source History

```
21          do
22          {
23              try
24              {
25                  Choice = in.nextInt();
26              }
27              catch(Exception e)
28              {
29                  Choice = 0;
                    in.next();
31              }
32
33              if (Choice < 1 || Choice > 4)
34                  System.out.print("Invalid choice.  Please reenter ")
35          }
36          while (Choice < 1 || Choice > 4);
37      }
38  }
39
```



Source History

```
21          do
22          {
23              try
24              {
25                  Choice = in.nextInt();
26              }
27              catch(Exception e)
28              {
29                  Choice = 0;
                    in.next();
31              }
32
33              if (Choice < 1 || Choice > 4)
34                  System.out.print("Invalid choice.  Please reenter ")
35          }
36          while (Choice < 1 || Choice > 4);
37      }
38  }
39
```

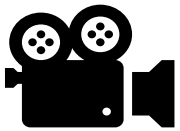
So now we have a menu choice. What do we do with it?

The value of `Choice` will be 1, 2, 3 or 4. This situation really says "USE A SWITCH"

```
switch(Choice)
{
    case 1 :

        break;
    case 2 :

        break;
    case 3 :

        break;
}
```

`Choice` of 4 is for exiting so we leave it out of the `switch` so that if 4 is chosen, the program completes.

Nothing was stated in the specification of the problem, but, for my own development and testing, I want the program to keep running until I choose to exit.



```
Source  History
10      public static void main(String[] args)
11      {
12          Scanner in = new Scanner(System.in);
13          int Choice = 0;
14
15          System.out.println("1. Encode file");
16          System.out.println("2. Decode file");
17          System.out.println("3. View file contents");
18          System.out.println("4. Exit");
19          System.out.print("\nEnter choice ");
20
21          do
22          {
23              try
24              {
25                  Choice = in.nextInt();
26              }
27              catch(Exception e)
28              {
29                  Choice = 0;
30                  in.nextLine();
```

Now that we have the menu setup, let's go back to what the menu does…

```
1. Encode file
2. Decode file
3. View file contents
4. Exit
```

Let's tackle menu item 3. To view the contents of a file, we will need to

ask for the file to be viewed
open that file
read that file
write the contents to the screen
close the file

ask for the file to be viewed

```
System.out.print("Enter file name ");
String IFilename = in.nextLine();
```

We declared a `Scanner` already

open that file

```
File IFH = new File(IFilename);
Scanner IF = new Scanner(IFH);
```

read that file/write the contents to the screen

```
while(IF.hasNextLine())
{
    System.out.println(IF.nextLine());
}
```

```
43
44              switch(Choice)
45              {
46                  case 1 :
47
48                      break;
49                  case 2 :
50
51                      break;
52                  case 3 :
53                      System.out.print("Enter file name ");
54                      String IFilename = in.nextLine();
55                      File IFH = new File(IFilename);
56                      Scanner IF = new Scanner(IFH);
57                      while(IF.hasNextLine())
58                      {
59                          System.out.println(IF.nextLine());
60                      }
61
62                      break;
63              }
```

When `nextInt()` read the menu choice from `stdin`, it left behind the <ENTER> key (newline).

Our `nextLine()` pulls that newline from `stdin` instead of asking for a filename.

Adding the `nextLine()` removes that newline from `stdin`.

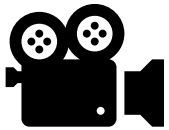
Now I want to add exception handling in case the file does not exist.

```
System.out.print("Enter file name ");
String IFilename = in.nextLine();
File IFH = new File(IFilename);

Scanner IF = new Scanner(IFH);
```

Line that throws exception if file does not exist

```
Exception in thread "main" java.io.FileNotFoundException: a (The system canno
t find the file specified)
        at java.base/java.io.FileInputStream.open0(Native Method)
        at java.base/java.io.FileInputStream.open(FileInputStream.java:213)
        at java.base/java.io.FileInputStream.<init>(FileInputStream.java:155)
        at java.base/java.util.Scanner.<init>(Scanner.java:639)
        at deleteme.DeleteMe.main(DeleteMe.java:57)
C:\Users\frenc\Documents\NetBeansProjects\DeleteMe\nbproject\build-impl.xml:1
355: The following error occurred while executing this line:
C:\Users\frenc\Documents\NetBeansProjects\DeleteMe\nbproject\build-impl.xml:9
61: Java returned: 1
BUILD FAILED (total time: 4 seconds)
```

```
57                      Scanner IF = new Scanner(IFH);
58                      while(IF.hasNextLine())
59                      {
60                          System.out.println(IF.nextLine());
61                      }
```



```
49                  case 2 :
50
51                      break;
52                  case 3 :
53                      in.nextLine();
54                      System.out.print("Enter file name ");
55                      String IFilename = in.nextLine();
56                      File IFH = new File(IFilename);
57                      Scanner IF = new Scanner(IFH);
58                      while(IF.hasNextLine())
59                      {
60                          System.out.println(IF.nextLine());
61                      }
62
63                      break;
64              }
65          }
66          while (Choice != 4);
67      }
68  }
69
```

Now that we have the menu setup, let's go back to what the menu does…

```
1. Encode file
2. Decode file
3. View file contents
4. Exit
```

We have coded menu option 3 and 4. Let's go back to the problem to see what menu option 1 and 2 should do.

The program should be able to encode and decode text files via this menu using a simple Caesar cipher using a left shift of 13.

Caesar cipher using a left shift of 13….

After googling that…

Caesar cipher using a left shift of 13….

Means shifting each letter of the input to the left by 13 characters.

So A (65) gets shifted to (65-13) which is 52 which is '4'.

z (122) gets shifted to (122-13) which is 109 which is 'm'.

Very simple; therefore, very easy to break but OK for our purposes.

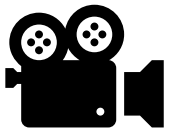
ASCII value	Character	ASCII value	Character	ASCII value	Character
000	^@	043	+	086	V
001	^A	044	,	087	W
002	^B	045	-	088	X
003	^C	046	.	089	Y
004	^D	047	/	090	Z
005	^E	048	0	091	[
006	^F	049	1	092	\
007	^G	050	2	093	]
008	^H	051	3	094	^
009	^I	052	4	095	_
010	^J	053	5	096	`
011	^K	054	6	097	a
012	^L	055	7	098	b
013	^M	056	8	099	c
014	^N	057	9	100	d
015	^O	158	:	101	e
016	^P	059	;	102	f
017	^Q	060	<	103	g
018	^R	061	=	104	h
019	^S	062	>	105	i
020	^T	063	?	106	j
021	^U	064	@	107	k
022	^V	065	A	108	l
023	^W	066	B	109	m
024	^X	067	C	110	n
025	^Y	068	D	111	o
026	^Z	069	E	112	p
027	^[	070	F	113	q
028	^\	071	G	114	r
029	^]	072	H	115	s
030	^^	073	I	116	t
031	^-	074	J	117	u
032	[space]	075	K	118	v
033	!	076	L	119	w
034	"	077	M	120	x
035	#	078	N	121	y
036	$	079	O	122	z
037	%	080	P	123	{
038	&	081	Q	124	|
039	'	082	R	125	}
040	(	083	S	126	~
041	)	084	T	127	DEL
042	*	085	U		

So for menu option 1, we need to encode the file. That means we need to

Steps

1. get the file name of the file to be encoded
2. get the file name of the file where we want to write the encoded version
3. read the file to be encoded
4. encode/shift each line
5. write each encoded line to the file where we want to write the encoded version

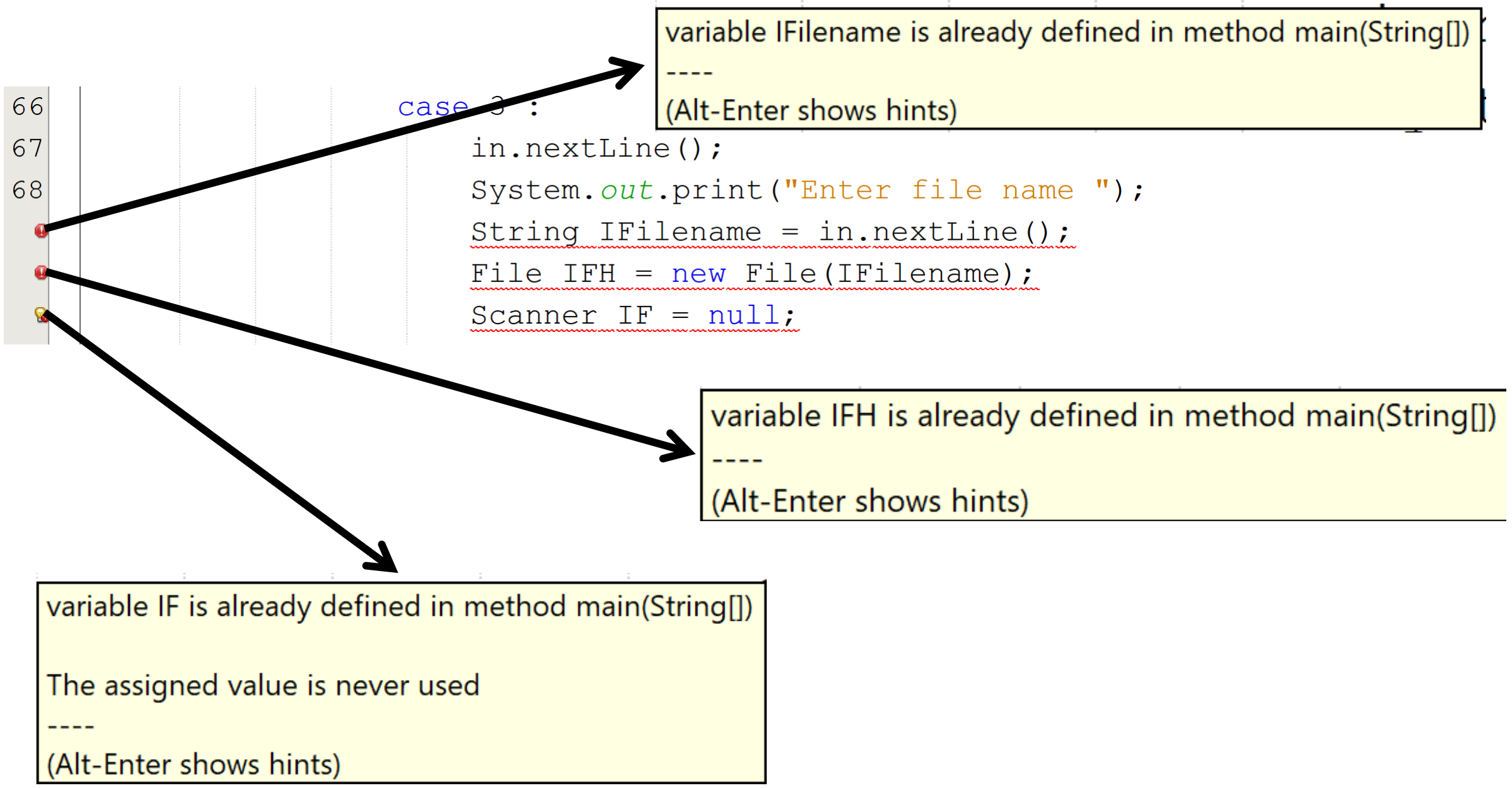
For Step 1, we have already written this code for menu option 3.

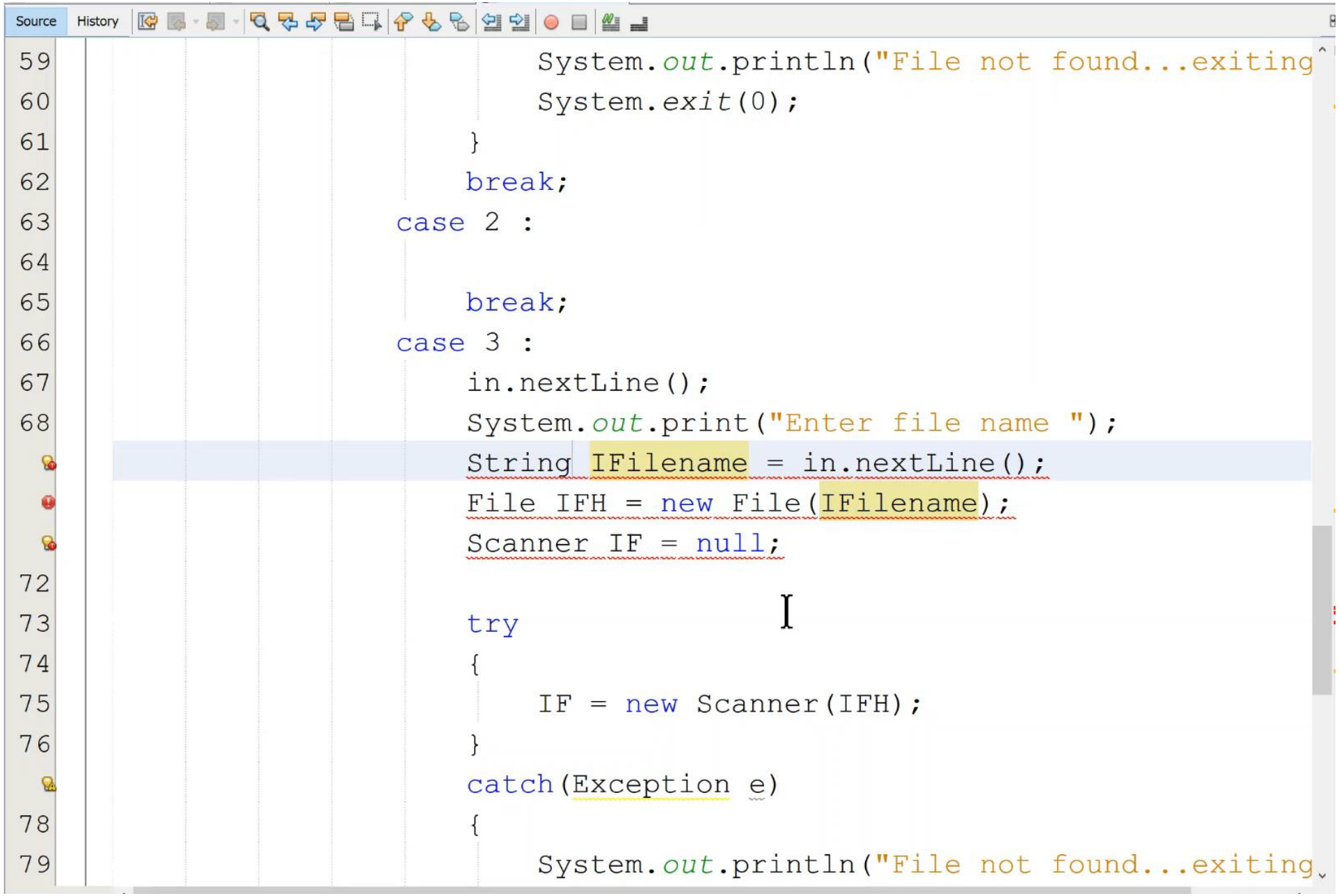


Source History

```
45                  {
46                      case 1 :
47
48                          break;
49                      case 2 :
50
51                          break;
52                      case 3 :
53                          in.nextLine();
54                          System.out.print("Enter file name ");
55                          String IFilename = in.nextLine();
56                          File IFH = new File(IFilename);
                            Scanner IF = null;
58
59                          try
60                          {
61                              IF = new Scanner(IFH);
62                          }
                            catch(Exception e)
64                          {
65                              System.out.println("File not found...exiting
```

Copying and pasting that code caused some errors to show up down in the case 3 code.



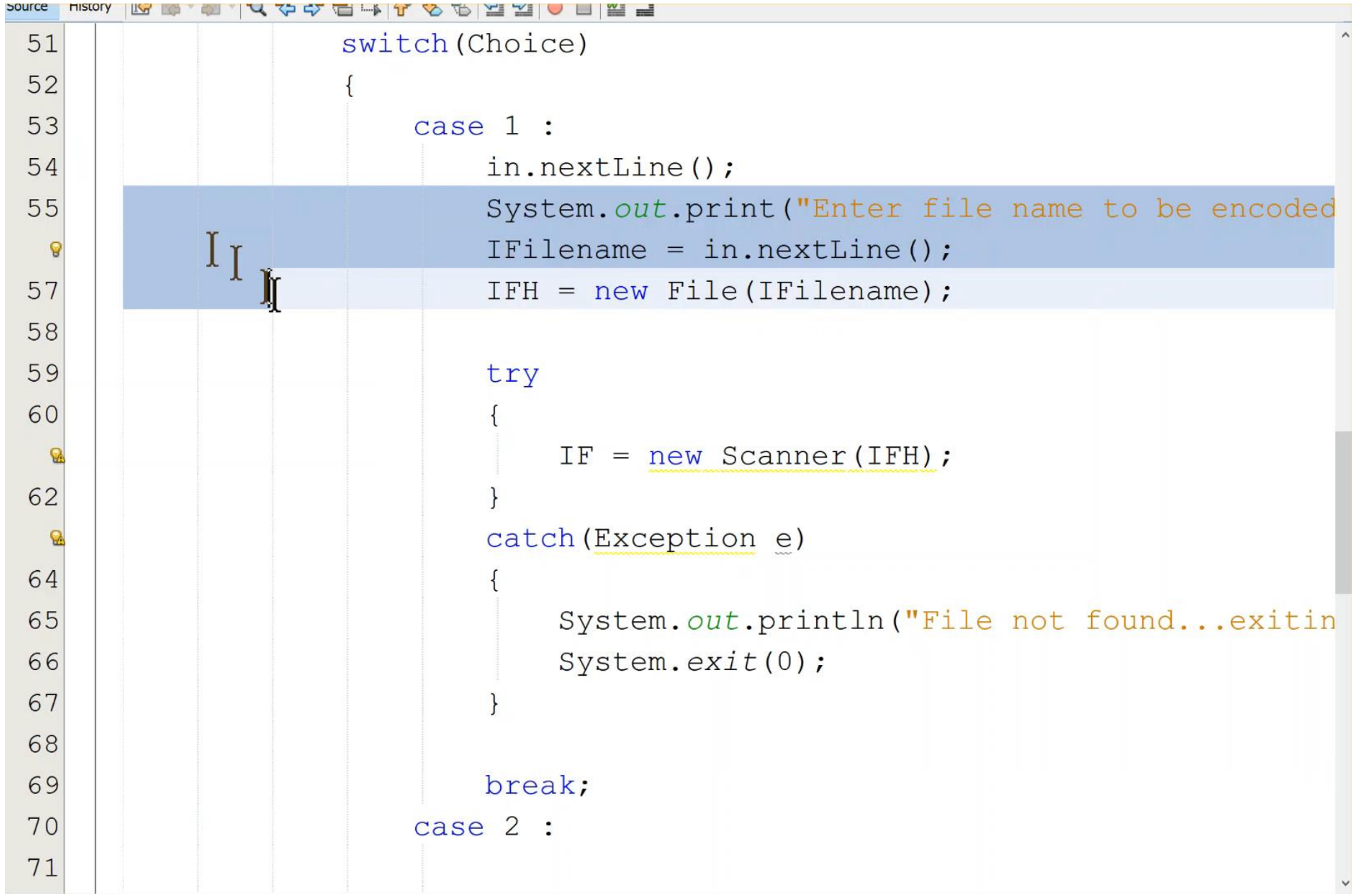

Source
History
59 System.out.println("File not found...exiting
60 System.exit(0);
61 }
62 break;
63 case 2 :
64
65 break;
66 case 3 :
67 in.nextLine();
68 System.out.print("Enter file name ");
String IFilename = in.nextLine();
File IFH = new File(IFilename);
Scanner IF = null;
72
73 try
74 {
75 IF = new Scanner(IFH);
76 }
catch(Exception e)
78 {
79 System.out.println("File not found...exiting

So for menu option 1, we need to encode the file. That means we need to

Steps

1. [redacted]
2. get the file name of the file where we want to write the encoded version
3. read the file to be encoded
4. encode/shift each line
5. write each encoded line to the file where we want to write the encoded version

For Step 2, the code we wrote for Step 1 is similar. We need to copy it and change it from opening a file for reading to opening a file for writing.


Source
History
switch(Choice)
{
case 1 :
in.nextLine();
System.out.print("Enter file name to be encoded
IFilename = in.nextLine();
IFH = new File(IFilename);
try
{
IF = new Scanner(IFH);
}
catch(Exception e)
{
System.out.println("File not found...exitin
System.exit(0);
}
break;
case 2 :

So for menu option 1, we need to encode the file. That means we need to

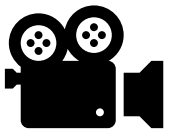
Steps

1. get the file name of the file to be encoded
2. get the file name of the file where we want to write the encoded version
3. read the file to be encoded
4. encode/shift each line
5. write each encoded line to the file where we want to write the encoded version

For Step 3, we have already written part of this code for menu option 3 that displayed the file

Source History

```
92
93                  try
94                  {
95                      IF = new Scanner(IFH);
96                  }
                    catch(Exception e)
98                  {
99                      System.out.println("File not found...exiting");
100                     System.exit(0);
101                 }
102                 while(IF.hasNextLine())
103                 {
104                    System.out.println(IF.nextLine());
105                 }
106
107                 break;
108             }
109     }
110     while (Choice != 4);
111
112
```



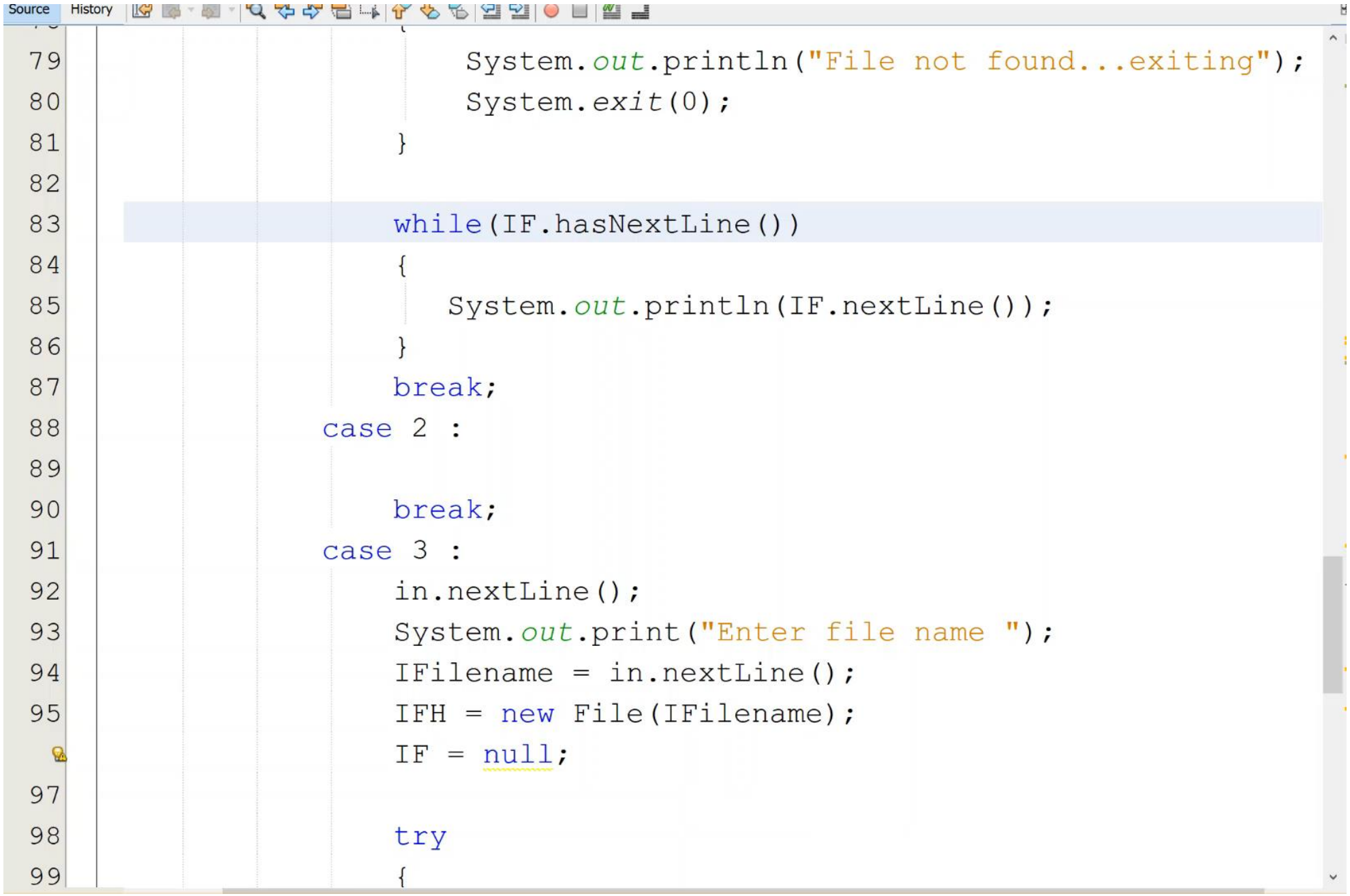
```
while(IF.hasNextLine())
{
    System.out.println(IF.nextLine());
}
```

This will read the entire file (the while) but we need to encode it and then print it to our newly opened output file.

Let's change this line

```
System.out.println(IF.nextLine());
```

to instead read the file line into a String and then write it to the output file.


Source
History
System.out.println("File not found...exiting");
System.exit(0);
}
while(IF.hasNextLine())
{
System.out.println(IF.nextLine());
}
break;
case 2 :
break;
case 3 :
in.nextLine();
System.out.print("Enter file name ");
IFilename = in.nextLine();
IFH = new File(IFilename);
IF = null;
try
{

```
while(IF.hasNextLine())
{
    FileLine = IF.nextLine();
    OF.println(FileLine);
}
```

Now we need to add the encoding step in between reading the line from the file and writing to the output file.

```
for (int i = 0; i < FileLine.length(); i++)
{
    OF.print((char)((int)FileLine.charAt(i) - 13));
}
OF.println();
```

```
83
84          while(IF.hasNextLine())
85          {
86              FileLine = IF.nextLine();
87              for (int i = 0; i < FileLine.length(); i++)
88              {
89                  OF.print((char)((int)FileLine.charAt(i) - 13));
90              }
91              OF.println();
92          }
93          OF.close();
94          break;
95      case 2 :
96
97          break;
98      case 3 :
99          in.nextLine();
100         System.out.print("Enter file name ");
101         IFilename = in.nextLine();
102         IFH = new File(IFilename);
            IF = null;
```

So for menu option 1, we need to encode the file. That means we need to

Steps

1. get the filename of the file to be encoded
2. get the filename of the file where we want to write the encoded version
3. read the file to be encoded
4. encode/shift each line
5. write each encoded line to the file that we want to write the encoded version

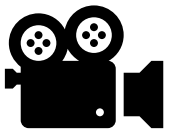
For Step 3, we have already written part of this code for menu option 3 that displayed the file

So for menu option 2, we need to decode the file. That means we need to

Steps

1. get the file name of the file to be decoded
2. read the file to decode
3. decode/shift each line
4. print the decoded lines to the screen

We have already most of this code for the other menu options.



```
Source  History
                                {
89                                  OF.print((char)((int)FileLine.charAt(i)
90                              }
91                              OF.println();
92                          }
93                          OF.close();
94                          break;
95                      case 2 :
96
97                          break;
98                      case 3 :
99                          in.nextLine();
                            System.out.print("Enter file name ");
101                         IFilename = in.nextLine();
102                         IFH = new File(IFilename);
                            IF = null;
104
105                         try
106                         {
107                             IF = new Scanner(IFH);
108                         }
                            catch(Exception e)
```