Kyle Phillips

Overflow HW

498-OS

Step 1 – 6 .JAVA

**2.**

**import** java.util.\*;

**public** **class** Overflow {

**static** **final** **int** ***INPUT\_SIZE (V)*** = 10;

**public** **static** **void** main(String[] args) {

**char**[] vals = **new** **char**[***INPUT\_SIZE***]; = 0 < i <= 9

Scanner scan = **new** Scanner(System.***in***);

String s1 = *getString*(scan); (V)

*copyVals*(s1, vals); <= 9

//vals will be equal to or less than 10 places when using method copyVals in this instance due to the value of vals at the time equaling the input size of 10. 0 < i <= 9

String sub = *getSubstring*(scan,vals); 0 < i <= 9

System.***out***.println("sub string: " + sub);

}

**public** **static** String getString(Scanner scan) {

System.***out***.print("Please type a string: ");

String s = scan.nextLine(); (V)

**return** s;

}

**public** **static** **void** copyVals(String s (V), **char**[] vals) {

**for** (**int** i = 0; i < s.length(); i++)(V) – limit of index is the same as the array, the length of inputted string s. 0 < i < s.length()-1 as arrays start at 0

{

vals[i] = s.charAt(i);

// array will be the length of the string s 0 < i < s.length() - 1

}

}

**public** **static** String getSubstring(Scanner scan (V), **char**[] vals) {

System.***out***.print("Starting point: ");

**int** start = scan.nextInt(); (V)

System.***out***.print("Ending point: ");

**int** end = scan.nextInt(); (V)

**char**[] newChars = *getChars*(start(V), end(V), vals);

**return** **new** String(newChars);

}

**public** **static** **char**[] getChars(**int** start(V), **int** end(V), **char**[] vals) {

**int** sz(V) = end(V) – start(V);

**char**[] result = **new** **char**[sz];

**for**(**int** i = 0; i < sz; i++)(V could go over) {

// the index is the same as the array result 0 < i < sz - 1

result[i] = vals[start + i];

// array results will be the length of the value of the end point minus the start point. 0 < i < sz -1

}

**return** result;

}

}

3.

* There is a buffer overflow in the main when using method copyVals if the length of the entered array is larger than 10 characters.
* Overflow with our sz if our end point is smaller than our start point.
* Overflow also with our getChar and getSubstring methods if the end point is smaller than our start point due to the fact that our getChar method will not work.
* Overflow in our main with the getSubString due to the overflow reasons being caused by our start and end variables listed above.
* Overflow if our starting point and ending point are out of our bounds created by the length of our input. Must be less than 10 in the case of our test program

4.

Please type a string: javaisthebest

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 10

at Overflow.copyVals(Overflow.java:33)

at Overflow.main(Overflow.java:16)

Please type a string: ilovejava

Starting point: 5

Ending point: 2

Exception in thread "main" java.lang.NegativeArraySizeException

at Overflow.getChars(Overflow.java:49)

at Overflow.getSubstring(Overflow.java:43)

at Overflow.main(Overflow.java:18)

Please type a string: ilovejava

Starting point: 15

Ending point: 25

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 15

at Overflow.getChars(Overflow.java:52)

at Overflow.getSubstring(Overflow.java:43)

at Overflow.main(Overflow.java:18)

5.

The easiest way to fix the potential of buffer overflow in general is to simply remove the chance for the user to create a problem. (Demonstrated in Overflow.JAVA)

* First add a if statement to test if the entered string is compatible with our array/fits and if not break and inform the user to enter a new string
* Then add a while loop to test if the starting and ending points inputted are inside the range of the array and if not inform the user to input a new end and start point and keep looping until the values are entered as desired.

If that’s too easy and we’re allowed to change the input size or simply remove it!!:

(Demonstrated in Overflow2.JAVA)

* We can simply remove the static input and create the array after the user inputted string to fit the entire string with no overflow.
* Then add a while loop to test if the starting and ending points inputted are inside the range of the array and if not ask the user to input a new end and start point and keep looping until the values are entered as desired.