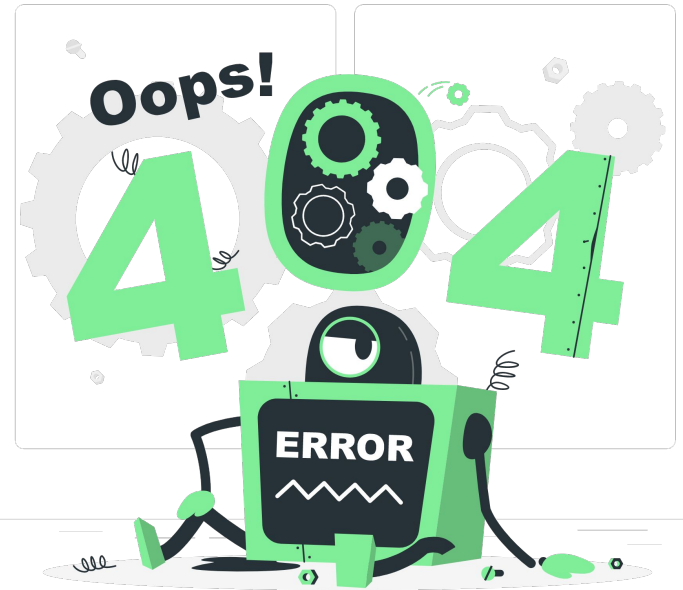


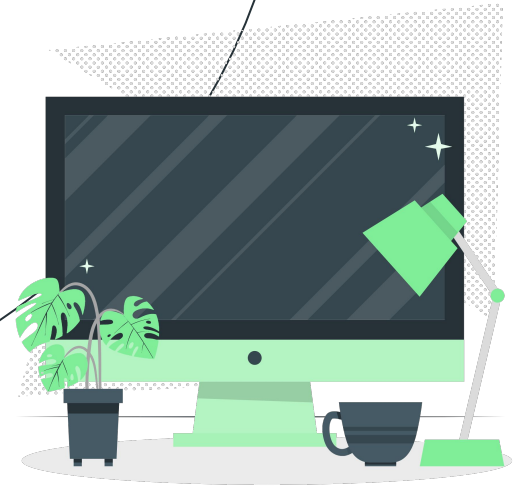
debugging 101

material adapted from CSE 142 20au



**“Bugs: If you don’t put them in,
you don’t have to take them out.”**

— CSE 2 124 Lab



what is debugging?

The process of **finding** and **removing** bugs from your code to make it run successfully.



game plan



01

debugging strategies

how to find bugs in your programs

02

common bugs

bugs that often appear in programs

03

finding bugs

practice finding bugs in small pieces of code

04

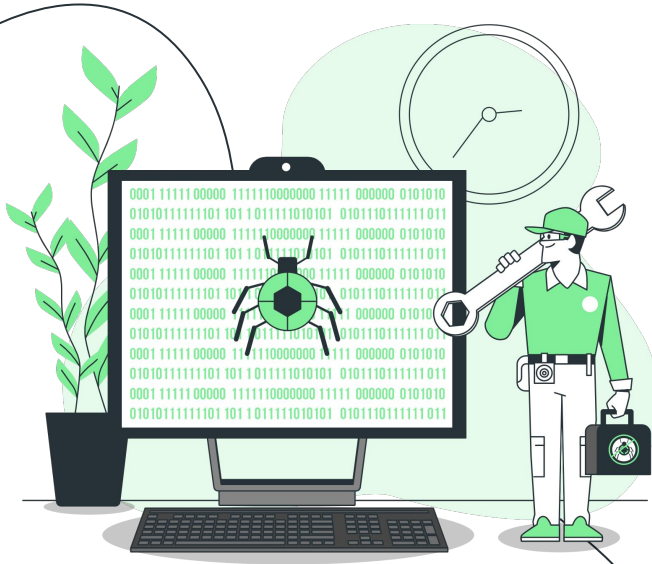
debugging a full program

practice debugging a full program

01 debugging strategies

including non-strategies and
effective strategies

material adapted from CSE
142 sp20, and CSE 332 18au



debugging tools

*In other words, these are strategies that you should **do** when debugging unexpected output.*

println debugging

- print out the intermediate states of the program
- fast, easy, effective

jGrasp debugger

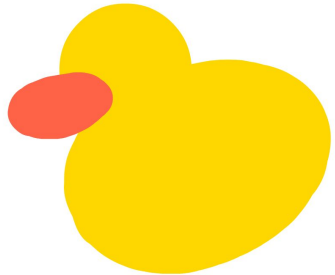
- Use jGrasp's built in debugging tool
- Lets you trace through your program's execution step by step
- A little work to learn but very useful

debugging strategies

*In other words, these are strategies that you should **do** when debugging unexpected output.*

Rubber Duck Debugging

- Grab a rubber duck (or another inanimate object, or friend) and explain your code to them
- Explain what your program does, line-by-line, and compare that to what it's supposed to do
- Sounds simple, but it works wonders



Take a Break

- Sometimes the solution will come to you when you're taking a walk, with friends, or watching a movie
- Taking a break gives your brain time to process information without stress

Ask for Help

- Come to **Support Hours**, explain the problem you're facing, what you've tried so far, and where you think the problem may be

debugging non-strategies

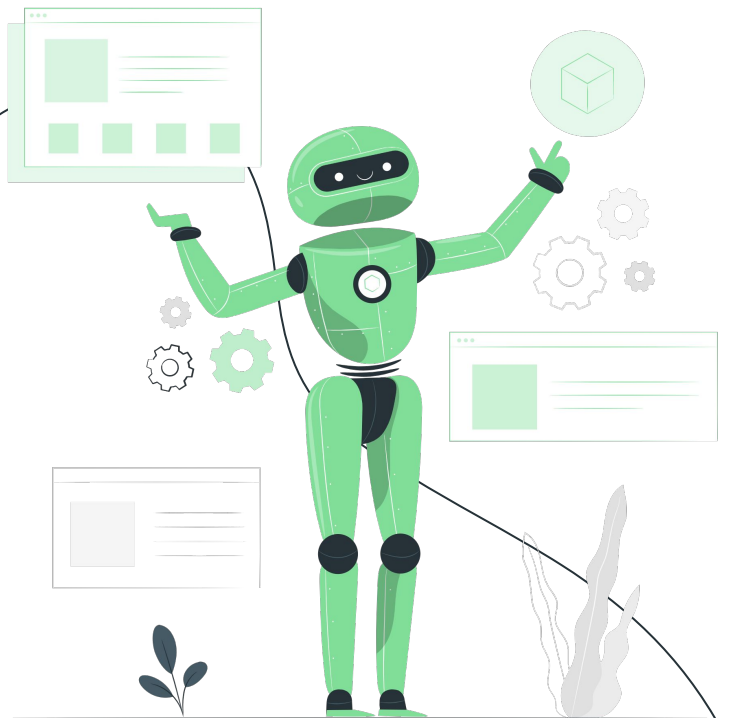
*In other words, these are strategies that you should **absolutely avoid** when debugging. They lead to extra frustration, often don't help you find the bug, and won't work as the programs get larger and more complicated.*

Shotgun Debugging

- Problem isn't clear, so let's just try every possible thing we can think of, like changing bounds in a loop
- Pls don't do this

Stare and Hope

- Stare at your code and hope you'll be able to see the bug
- An expectation that your brain is to run the program in your head, so why not use the computer to your advantage



02 common compiler & runtime bugs

<https://courses.cs.washington.edu/courses/cse142/21sp/files/debugging.pdf>

03

finding bugs

practice addressing some
common bugs



guided bug #1

```
public class Avatar {
    public static void main(String[] args) {
        int x = 0;
        if (x == 1) {
            System.out.println("Toph is awesome");
        } else
            System.out.println("Toph is cool");
        }
    }
}
```

After compiling, I receive this bug- what's going on?

```
Avatar.java:10: error: class, interface, or enum expected
}
^
1 error
```

guided bug #1

```
public class Avatar {
    public static void main(String[] args) {
        int x = 0;
        if (x == 1) {
            System.out.println("Toph is awesome");
        } else
            System.out.println("Toph is cool");
        }
    }
}
```

After compiling, I receive this bug- what's going on?

```
Avatar.java:10: error: class, interface, or enum expected
}
^
1 error
```

Double check that your program has the right # of opening { AND closing } curly braces!

In this program:

- { = 3
- } = 4

closing != # opening

Too many closing braces :(

guided bug #1 solution

```
public class Avatar {  
    public static void main(String[] args) {  
        int x = 0;  
        if (x == 1) {  
            System.out.println("Toph is awesome");  
        } else  
            System.out.println("Toph is cool");  
        }  
    }  
}
```

```
public class Avatar {  
    public static void main(String[] args) {  
        int x = 0;  
        if (x == 1) {  
            System.out.println("Toph is awesome");  
        } else {  
            System.out.println("Toph is cool");  
        }  
    }  
}
```

Add an opening brace

guided bug #2

```
public class Avatar {  
    public static void main(String[] args) {  
        int x = 0;  
        if (x == 1) {  
            System.out.println("Toph is awesome");  
        } else if (x == 0) {  
            System.out.println("Toph is cool");  
        }  
    }  
}
```

After compiling, I receive this bug- what's going on?

```
Avatar.java:10: error: reached end of file while parsing
```

```
}  
^
```

guided bug #2

```
public class Avatar {  
    public static void main(String[] args) {  
        int x = 0;  
        if (x == 1) {  
            System.out.println("Toph is awesome");  
        } else if (x == 0) {  
            System.out.println("Toph is cool");  
        }  
    }  
}
```

After compiling, I receive this bug- what's going on?

```
Avatar.java:10: error: reached end of file while parsing
```

```
}  
^
```

Similar issue as before - double check that your program has the right # of opening { AND closing } curly braces!

In this program:

- { = 4
- } = 3

closing != # opening

Too many closing braces :(

guided bug #2

solution

```
public class Avatar {
    public static void main(String[] args) {
        int x = 0;
        if (x == 1) {
            System.out.println("Toph is awesome");
        }
        else if (x == 0) {
            System.out.println("Toph is cool");
        }
    }
}
```

Add a closing brace

```
public class Avatar {
    public static void main(String[] args) {
        int x = 0;
        if (x == 1) {
            System.out.println("Toph is awesome");
        }
        else if (x == 0) {
            System.out.println("Toph is cool");
        }
    }
}
```


guided bug #3

```
public class Avatar {  
    public static void main(String[] args) {  
        String appa = "fluffy flying bison";  
    }  
  
    public static void aang() {  
        System.out.println(appa);  
    }  
}
```

```
Avatar.java:8: error: cannot find symbol  
        System.out.println(appa);  
                           ^  
symbol:   variable appa  
location: class Avatar  
1 error
```

After compiling, I receive this bug- what's going on?

I've declared **appa** in my main method, so, this should work, right?

Why can't **aang()** find **appa**? (hehe)

guided bug #3

```
public class Avatar {
    public static void main(String[] args) {
        String appa = "fluffy flying bison";
    }

    public static void aang() {
        System.out.println(appa);
    }
}
```

```
Avatar.java:8: error: cannot find symbol
        System.out.println(appa);
                           ^
    symbol:   variable appa
    location: class Avatar
1 error
```

If a method can't find a symbol, that means that variable isn't in the scope of that method.

So, how do we get `aang()` to know about `appa`, the variable that only exists in the main method?

guided bug #3

solution

```
public class Avatar {  
    public static void main(String[] args) {  
        String x = "fluffy flying bison";  
    }  
  
    public static void aang() {  
        System.out.println(appa);  
    }  
}
```

```
public class Avatar {  
    public static void main(String[] args) {  
        String x = "fluffy flying bison";  
        aang(x);  
    }  
  
    public static void aang(String appa) {  
        System.out.println(appa);  
    }  
}
```

Add 'appa' as a
parameter



guided bug #4

```
public class Avatar {  
    public static void main(String[] args) {  
        String s = aang();  
    }  
  
    public static String aang() {  
        return 2;  
    }  
}
```

After compiling, I receive this bug- what's going on?

```
Avatar.java:8: error: incompatible types: int cannot be converted to String  
    return 2;  
           ^
```

1 error

guided bug #4

```
public class Avatar {  
    public static void main(String[] args) {  
        String s = aang();  
    }  
  
    public static String aang() {  
        return 2;  
    }  
}
```

After compiling, I receive this bug- what's going on?

```
Avatar.java:8: error: incompatible types: int cannot be converted to String  
        return 2;  
                ^
```

1 error

The data that a method returns MUST match the return type of the method

Type of data being returned = **int**
Return type of **aang()** = **String**

Problem: an **int** cannot be converted into a **String**

guided bug #4

solution

```
public class Avatar {
    public static void main(String[] args) {
        int s = aang();
    }

    public static int aang() {
        return 2;
    }
}
```

```
public class Avatar {
    public static void main(String[] args) {
        String s = aang();
    }

    public static String aang() {
        return "2";
    }
}
```

Change to a string

Two ways to solve this:

1. Change the 2 into "2" (therefore making it a string)
2. Or, change the return type altogether of the aang() method to int instead of string. The type of data being caught will also need to change to int (see main method)

04 debugging a full program

Let's take a look at the `BuggyRoulette.java` program

