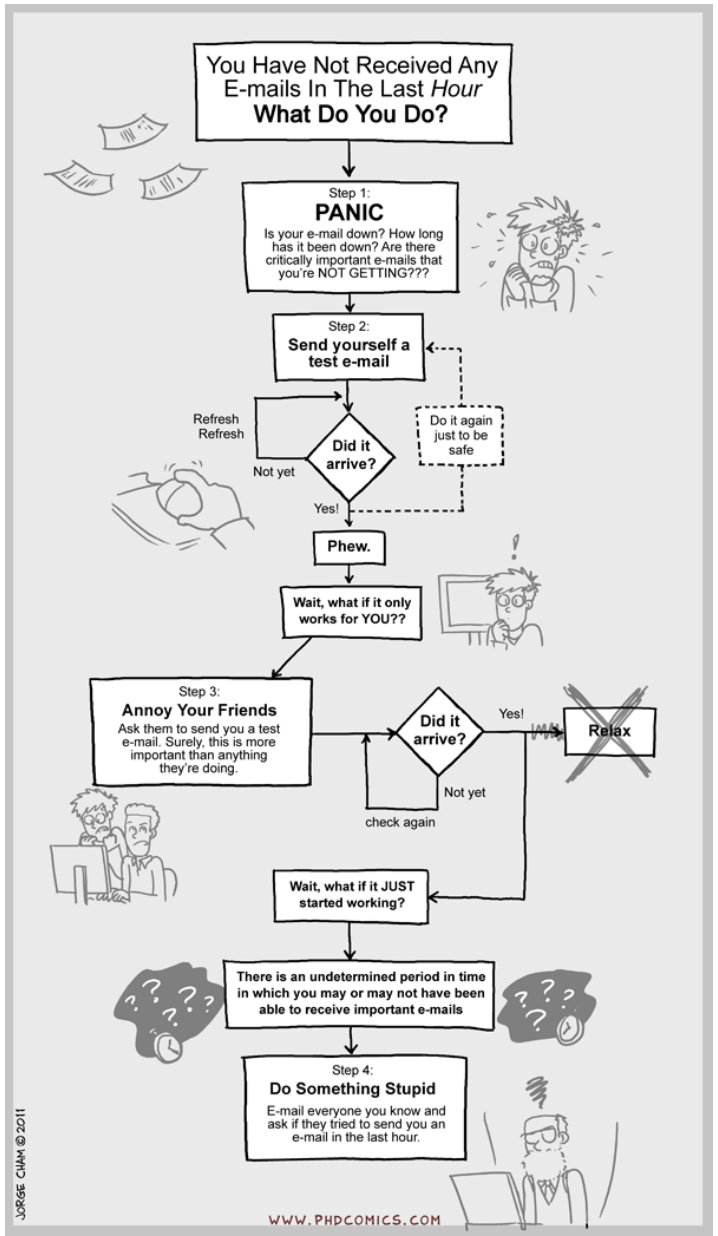
CS 161 Lecture 4 – More Conditionals – 19 January 2017



if/else if/else

```
if (<expression>) {
        <statement>
}
…
else if (<expression>) {
        <statement>
}
…
else {
        <statement>
}
```

Relational and Logical Operators

- == Equality
- >= Greater than or equal to
- <= Less than or equal to
- > Greater than
- < Less than
- != Not equal to
- || OR
- && AND
- ! NOT

Example

- if(1+2)
- if(2-4)
- if(2-2)
- if(4==4)
- if((2+1) == 4)
- if(4.1 == 4)
- if(3 <= 4)
- if(4 >= 4)
- if(3.5 > 4)
- if(4 < 4)
- if(3+2*2 > 9)
- if((3+2)*2 > 9)

Examples Continued

- AND: if((1>2) && (2<5))
- OR: if((1>2) || (2<5))
- NOT: if(!(1>2) && (2<5))

p	q	p && q	p || q	!p
T	T	T	T	F
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

if/else Example

```
if (x > y) {
        cout << “X is greater than Y” << endl;
}
else {
        cout << “Y is less than X” << endl;
}
//Are these print statements always true?
```

if/else if/else Example

```
if (x > y) {
        cout << “X is greater than Y” << endl;
}
else if (x == y) {
        cout << “X is equal to Y” << endl;
}
else {
        cout << “Y is less than X” << endl;
}
```

Nested Decisions

```
if (confused_on_class_procedure == True) {
        if (procedure_in_syllabus == False) {
                cout << “Email TA or Shannon” << endl;
        } else {
                bool still_confused = read_syllabus();
                if (still_confused == True) {
                        cout << “Email TA or Shannon” << endl;
                } else {
                        cout << “Good job!” << endl;
                }
        }
}
```

Notes on Scope

- Block Scope: variables declared exist until the end of the block
- Blocks = {}

```
int main () {
        int a = 0;
        if (a == 0) {
                int b = a; //a still exists in here
        }
        b = 1; //error: b does not exist out here
        return 0;
}
```

Alternative: Switch Statements

```
switch (<expression>) {
        case <const-expression>:
                <statement>;
                …
                break;
        case <const-expression>:
                <statement>;
                …
                break;
        default:
                <statement>;
                …
                break;
}
```

Switch Statement Details

- Tests equality: <expression> == <const-expression>
  - If it’s true, execute that code
- Need to have break statements otherwise it will fall through and execute everything else
- Default case is optional but a good idea

Demo

Feedback

https://tinyurl.com/yau3323k