CSc 120

Introduction to Computer Programming II

CODE EXAMPLES 01 ☺ ☹

Example 1 ☹

```
def grid_is_square(arglist):
    length= len(arglist)
    i = 0
    for i in range(length):
        x = 0
        if arglist[i][x] == arglist[x][i]:
           return True
       elif arglist[i][x] != arglist[x][i]:
          return False
```

```
def grid_is_square(arglist):
    length= len(arglist)
    i = 0
    for i in range(length):
        x = 0
        if arglist[i][x] == arglist[x][i]:
            return True
        elif arglist[i][x] != arglist[x][i]:
            return False
```

```
if BoolExpr:
    x = True
else:
    x = False
```

is equivalent to:

```
x = BoolExpr
```

```
def grid_is_square(arglist):
    length= len(arglist)
    i = 0
    for i in range(length):
        x = 0
        return arglist[i][x] == arglist[x][i]
```

Simplifying out the Boolean expression makes the bug in the code easier to see

```
def grid_is_square(arglist):
    length= len(arglist)
    i = 0
    for i in range(length):
        x = 0
        return arglist[i][x] == arglist[x][i]
```

this initialization has no effect

```
def grid_is_square(arglist):
    length= len(arglist)
    i = 0
    for i in range(length):
        x = 0
        return arglist[i][x] == arglist[x][i]
```

we don't need this

```
def grid_is_square(arglist):
    length= len(arglist)
    for i in range(length):
        return arglist[i][0] == arglist[0][i]
```

returns on the first iteration

Example 2 ☺

```
def concat_elements(list, startpos, stoppos):
    result = ""
    if (startpos < 0):
        startpos = 0
    if (stoppos > len(list)):
        stoppos = len(list)
    if (startpos > stoppos):
        return ''
    a = list[startpos:stoppos+1]

    result = ''.join(a)

    return result
```

```
def concat_elements(list, startpos, stoppos):
    result = ""
    if (startpos < 0):
        startpos = 0
    if (stoppos > len(list)):
        stoppos = len(list)
    if (startpos > stoppos):
        return ''
    a = list[startpos:stoppos+1]

    result = ''.join(a)

    return result
```

```
startpos = max(startpos, 0)
```

```
stoppos = min(stoppos,
              len(list))
```

```
def concat_elements(list, startpos, stoppos):
    result = ""
    startpos = max(startpos,0)
    stoppos = min(stoppos, len(list))

    if (startpos > stoppos):
        return ''
    a = list[startpos:stoppos+1]

    result = ''.join(a)

    return result
```

This is not really needed

```
def concat_elements(list, startpos, stoppos):
   new = ""
   for i in range (max(0, startpos),
                         min( len(list), stoppos + 1)):
      new += list [i]

   return new
```