John Romero Programming Proverbs

- 5. “We are our own best testing team and should never allow anyone else to experience bugs or see the game crash. Don’t waste others’ time. Test thoroughly before checking in your code.”

- John Romero, “The Early Days of Id Software - John Romero @ WeAreDevelopers Conference 2017”

Adding auto lights to chisel

- propose to add two options to `txt2pen.py`
  - `-l` to enable auto lights
  - `-f num` to change the default frequency of lights (default is every five squares)

Code changes to chisel/python/txt2pen.py

```
inputFile = None
defines = {}
verbose = False
debugging = False
autoLights = False
floor = []
rooms = {}
maxx, maxy = 0, 0
doorValue, wallValue, emptyValue = 0, -1, -2
versionNumber = 0.1
lightFrequency = 5
```

- notice the new global variables `autoLights` and `lightFrequency`

Code changes to chisel/python/txt2pen.py

```
def usage (code):
    print "Usage: txt2pen [-dhlvV] [-f frequency] [-o outputfile] inputfile"
    print "  -d debugging"
    print "  -h help"
    print "  -l automatic lighting"
    print "  -f frequency    (every frequency squares place a light)"
    print "  -V verbose"
    print "  -v version"
    print "  -o outputfile name"
    sys.exit (code)
```

Code changes to chisel/python/txt2pen.py

```
class roomInfo:
    def __init__ (self, w, d):
        self.walls = w
        self.doors = d
        self.doorLeadsTo = []
        self.monsters = []
        self.weapons = []
        self.ammo = []
        self.lights = []
        self.autoLights = []
        self.worldspawn = []
```

Code changes to chisel/python/txt2pen.py

```
def handleOptions ():
    global debugging, verbose, outputName, autoLights, lightFrequency

    outputName = None
    try:
        optlist, l = getopt.getopt(sys.argv[1:], ’:df:hlo:vV’)
        for opt in optlist:
            if opt[0] == ’-d’:
                debugging = True
            elif opt[0] == ’-h’:
                usage (0)
            elif opt[0] == ’-l’:
                autoLights = True
            elif opt[0] == ’-f’:
                lightFrequency = int (opt[1])
            elif opt[0] == ’-o’:
                outputName = opt[1]
etc...
```

New function checkLight

-
  ```
  #
  #   checkLight - add a mid light if lightCount == lightFrequency
  #

  def checkLight (position, lightList, lightCount):
      if lightCount == lightFrequency:
          li = light ()
          li.settype ('MID')
          lightList += [position + [li]]
          lightCount = 0
      else:
          lightCount += 1
      return lightList, lightCount
  ```

- which is called from your `introduceLights`

txt2pen changes

```
def generateRoom (roomNo, position, mapGrid, start, lineNo):
    global verbose, rooms, debugging

    inside = position
    position = moveBy (position, [-1, -1], mapGrid)
    if debugging:
        print ("top left is", position)
    start = position
    walls, doors = scanRoom (start, position, mapGrid, [], [])
    if debugging:
        print(walls)
    rooms[roomNo] = roomInfo (walls, doors)
    rooms[roomNo].autoLights += introduceLights (position, mapGrid, [], [])
    rooms[roomNo].inside = inside
```

function printRoom changes

```
etc...
    outputFile = printMonsters (rooms[roomNo].monsters, outputFile)
    outputFile = printAmmo (rooms[roomNo].ammo, outputFile)
    outputFile = printWeapons (rooms[roomNo].weapons, outputFile)
    if autoLights and (rooms[roomNo].lights == []):
        outputFile = printLights (rooms[roomNo].autoLights, outputFile)
    else:
        outputFile = printLights (rooms[roomNo].lights, outputFile)
    outputFile = printSpawnPlayer (rooms[roomNo].worldspawn, outputFile)
etc...
```

function printRoom changes

- you need to complete `introduceLights` to make these changes take effect