Challenge 01 - Defuse the Bomb!
Description

To disarm the bomb you have to cut some wires.

These wires are either white, black, purple, red, green or orange.

There can be multiple wires with the same color. These instructions are for one wire at a time. Once you cut a wire you can forget about the previous ones.

The rules for disarming are simple:

```
If you cut a white cable you can't cut white or black cable.
If you cut a red cable you have to cut a green one
If you cut a black cable it is not allowed to cut a white, green or orange one
If you cut a orange cable you should cut a red or black one
If you cut a green one you have to cut a orange or white one
If you cut a purple cable you can't cut a purple, green, orange or white cable
```

If you have anything wrong in the wrong order, the bomb will explode.

Formal Inputs & Outputs, Input description
You will receive a sequence of wires that were cut in that order and you have to determine if the person was successful in disarming the bomb or that it blew up.

Input 1
```
white
red
green
white
```

Input 2
```
white
orange
green
White
```

Output description
Whether or not the bomb exploded

Output 1
```
"Bomb defused"
```

Output 2
```
"Boom"
```

Challenge 02 - Fallout Hacker

Description

The popular video games Fallout 3 and Fallout: New Vegas have a computer "hacking" minigame where the player must correctly guess the correct password from a list of same-length words. Your challenge is to implement this game yourself.

The player has only 4 guesses and on each incorrect guess the computer will indicate how many letter positions are correct.

For example, if the password is MIND and the player guesses MEND, the game will indicate that 3 out of 4 positions are correct (M_ND). If the password is COMPUTE and the player guesses PLAYFUL, the game will report 0/7. While some of the letters match, they're in the wrong position.

Ask the player for a difficulty (very easy, easy, average, hard, very hard), then present the player with 5 to 15 words of the same length. The length can be 4 to 15 letters. More words and letters make for a harder puzzle. The player then has 4 guesses, and on each incorrect guess indicate the number of correct positions.

Here's an example game:

```
Difficulty (1-5)? 3
SCORPION
FLOGGING
CROPPERS
MIGRAINE
FOOTNOTE
REFINERY
VAULTING
VICARAGE
PROTRACT
DESCENTS
Guess (4 left)? migraine
0/8 correct
Guess (3 left)? protract
2/8 correct
Guess (2 left)? croppers
8/8 correct
You win!
```

You can draw words from the dictionary file: enable1.txt. Your program should completely ignore case when making the position checks.

There may be ways to increase the difficulty of the game, perhaps even making it impossible to guarantee a solution, based on your particular selection of words. For example, your program could supply words that have little letter position overlap so that guesses reveal as little information to the player as possible.

CHALLENGE 03 - Procedural Music Generation Description

This is a more open-ended challenge: write a program to generate music. The goal is to create a piece of music that has never been written down or heard before.

You can use whatever methods you want, including using existing music as training data or input. But you should not just apply straightforward manipulations to existing music. One common technique is to train a Markov chain model on existing music. Other techniques are perfectly valid too, such as cellular automata or neural networks.

Make it as short or as long as you like. Even 16 beats is not easy.

Training/input data

For your convenience here are a few pieces of music in an easy-to-use format, which you can listen to using the tool below. You don't have to use these pieces: you can use other pieces, or no inputs at all.

Mozart: Rondo Alla Turca

Mozart: Eine Kleine Nachtmusik

Debussy: Clair de Lune

Beethoven: Für Elise

Each line in this format specifies a note with three values: pitch, starting time, and duration. The pitch number is the MIDI note number, where middle C = 60. The starting time and duration are both in units of beats. For example, the line:

```
37 8.5 0.5
```

means to play the note C#3 (corresponding to a pitch number of 37) for 0.5 beats, starting at beat 8.5.

You can use this format for your output too, or use any other format you like (including audio formats). Of course, it's more fun if you can listen to it one way or another!

Listening and sharing

Use this JavaScript page that converts the above format into generated sounds. If you use the same format, paste your output there and hit play to listen.

You might also consider, instead of randomly choosing notes, basing your selection of notes on real-world scientific data.

Credit to /r dailyprogrammer, if you enjoyed these definitely find more there!