Randomized Graphing

Attempts at randomizing graphing.

Ways to generate randomized graphs (Still being worked on)

Using sageOutput environment

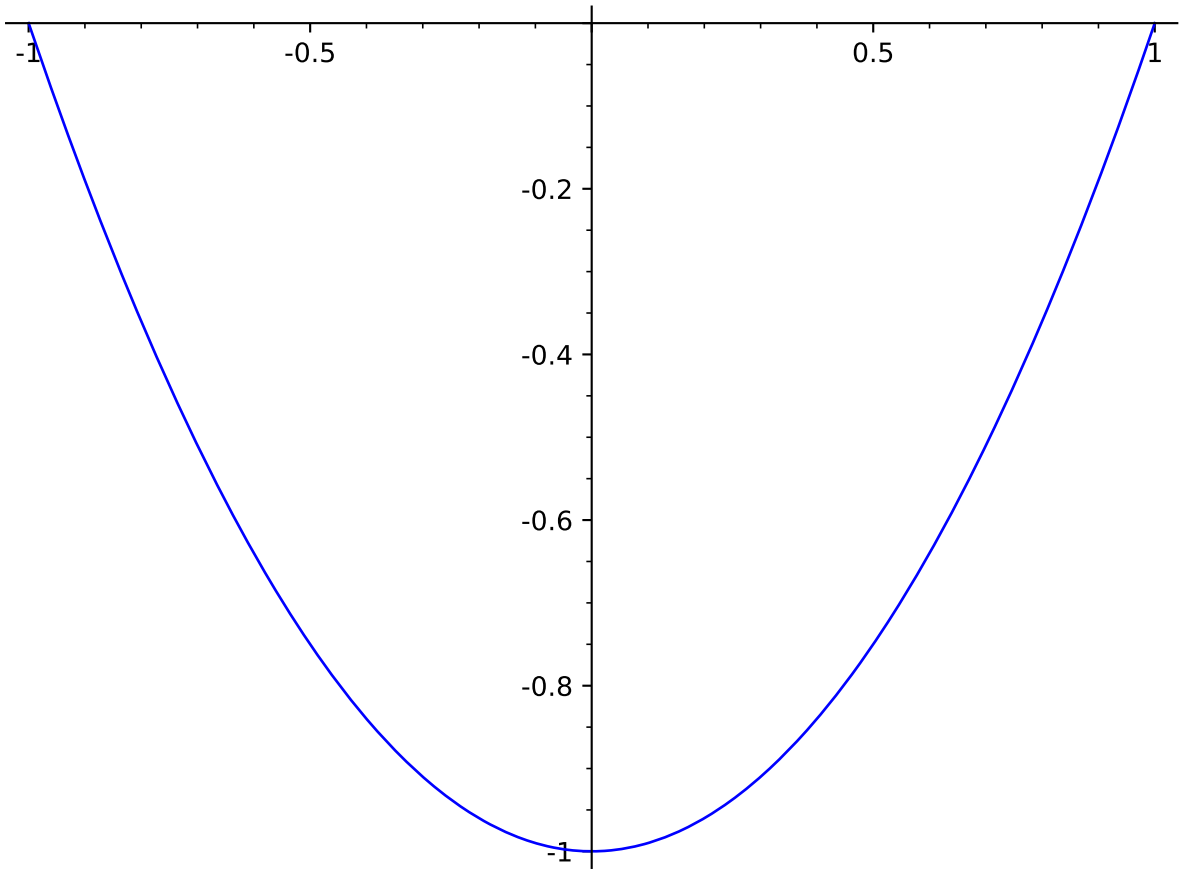
SAGE-Output

```
def RandInt(a,b):
    """ Returns a random integer in [‘a‘,‘b‘]. Note that ‘a‘ and ‘b‘ should be integers themselves to avoid u
    """
    return QQ(randint(int(a),int(b)))
    # return choice(range(a,b+1))

def NonZeroInt(b,c, avoid = [0]):
    """ Returns a random integer in [‘b‘,‘c‘] which is not in ‘av‘.
        If ‘av‘ is not specified, defaults to a non-zero integer.
    """
    while True:
        a = RandInt(b,c)
        if a not in avoid:
            return a

p1temp1 = 'temp'

p1temp2 = x^2 + NonZeroInt(-2,2)
plot(p1temp2,(x,-3,3))
```

Note that everything inside a sageOutput environment is essentially locked into it's own scope - no using variables from in there to populate environments elsewhere on the page (e.g. no using the same random number to define a function in a problem, and display a function on the page - that I've found).

Above is (probably) NOT a graph of $x^2 - 1$ for example.

Learning outcomes:



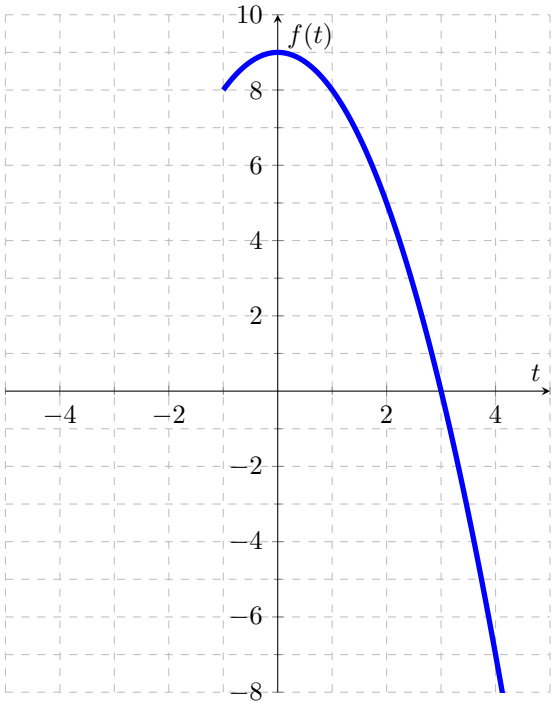
temp

temp

Using Desmos

Desmos graph command seems.... broken maybe?

Let's make sure tikz still works:


10
8
6
4
2
−2
−4
−6
−8
−4
−2
2
4
$f(t)$
t