In [2]:

```
from __future__ import print_function, division

%matplotlib inline

import thinkdsp
import thinkplot
import numpy as np
import math

import warnings
warnings.filterwarnings('ignore')

from IPython.html.widgets import interact, interact_manual, fixed
from IPython.html import widgets
from IPython.display import display

PI2 = 2 * math.pi#원주율
```

In [3]:

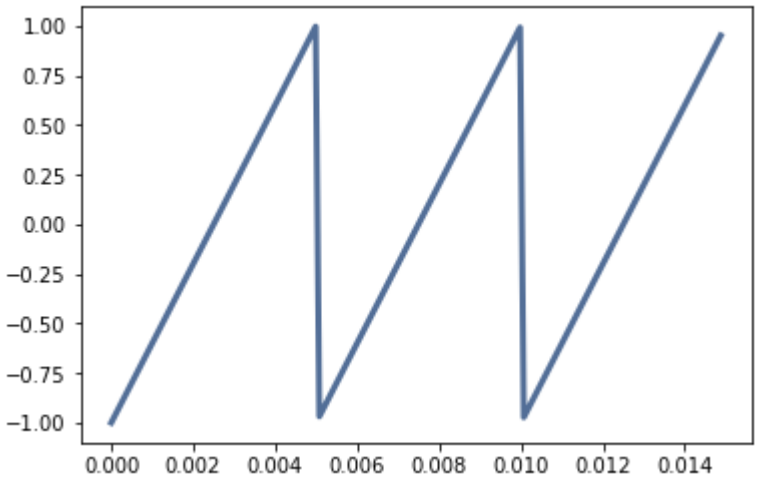
```
class SawtoothSignal(thinkdsp.Sinusoid):#2.2번


    def evaluate(self, ts):

        cycles = self.freq * ts + self.offset / PI2
        frac, _ = np.modf(cycles)# _: 값을 무시한다.
        ys = thinkdsp.normalize(thinkdsp.unbias(frac), self.amp)# 중간점의 y좌표를 0으로 만든다.
        return ys
```

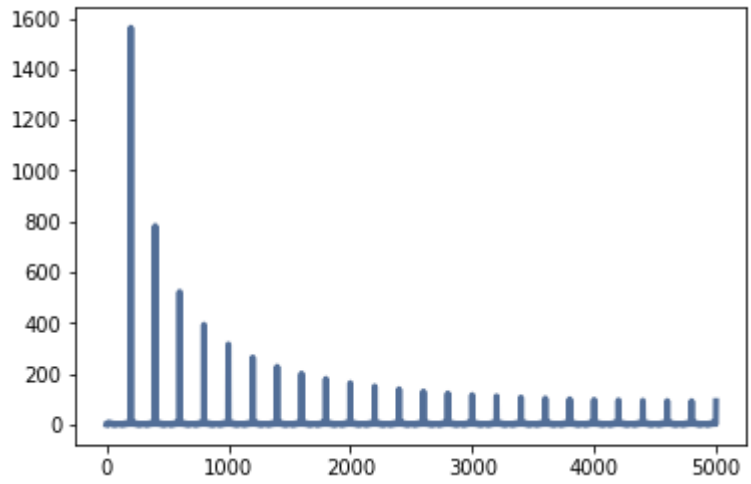
In [5]:

```
signal=thinkdsp.SawtoothSignal(200)
signal.plot()#sawtooth 그래프 그리기
```



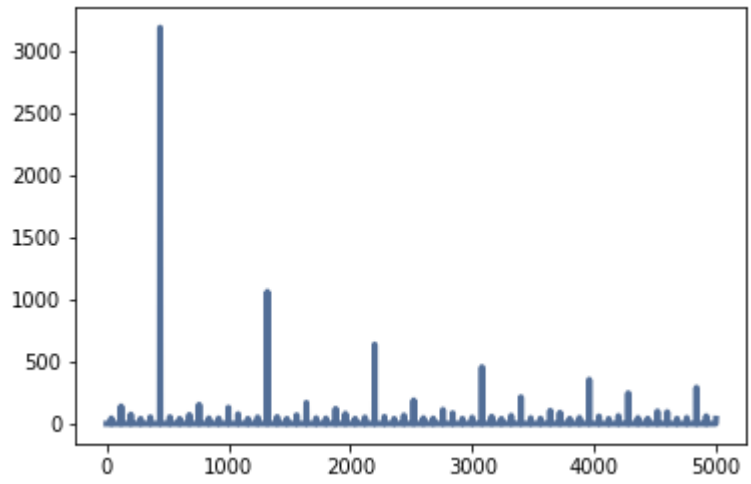
In [6]:

```
wave=signal.make_wave(duration=0.5, framerate=10000)
spectrum=wave.make_spectrum()
spectrum.plot()
#스펙트럼 만들기!!!
```



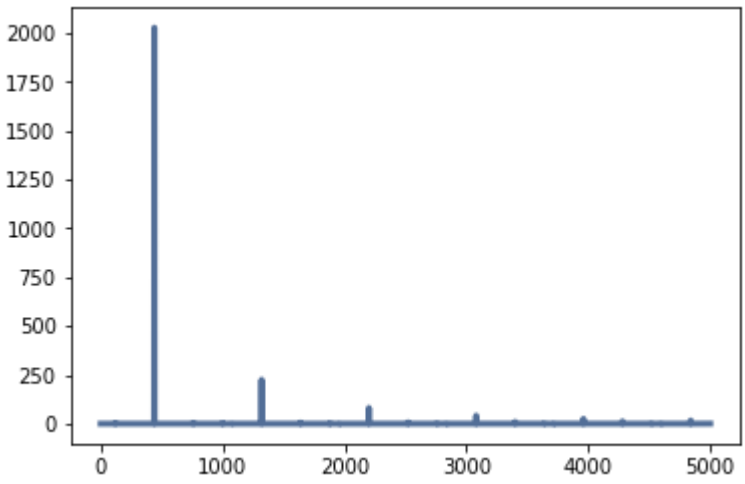
In [7]:

```
square = thinkdsp.SquareSignal(amp=1).make_wave(duration=0.5, framerate=10000)
square.make_spectrum().plot()
```



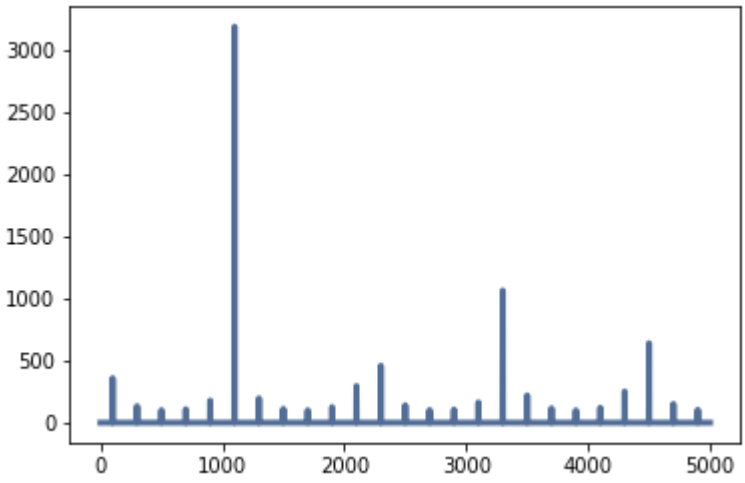
In [8]:

```
triangle = thinkdsp.TriangleSignal(amp=1).make_wave(duration=0.5, framerate=10000)
triangle.make_spectrum().plot()
#세그래프의 비교는 똑같이 스펙트럼을 그려서 fundamental freq를 확인한다.
```



In [9]:

```
#2.3번
square = thinkdsp.SquareSignal(1100).make_wave(duration=0.5, framerate=10000)
square.make_spectrum().plot()
#1100Hz가 기저 주파수이고 초당 10000프레임이 셈플링되는 signal 입니다.
```



In [10]:

```
square.make_audio()
```

Out[10]:

In [ ]:

```
#No!!! aliasing: 위상은 다른데 샘플이 같다.
#따라서 harmonics가 깨지므로 들을 수가 없었습니다.
```