1. Uniform({13, ..., 66})

In [5]:
```
import numpy as np
import scipy.stats as sps
from tqdm import tqdm
import matplotlib.pyplot as plt
import pandas as pd
%matplotlib inline
```

$$T_n = \frac{\sum_{i=1}^{n} X_i - nEX_1}{\sqrt{nDX_1}}$$

$EX_1 = \frac{a+b}{2} = \frac{13+66}{2} = 39.5$

$DX_1 = \frac{(b-a)^2}{12} = 234.0833$

In [2]:
```
variation = ((66 - 13) ** 2) / 12
variation
```

Out[2]:
```
234.08333333333334
```

In [3]:
```
matozh = 39.5
```

In [4]:
```
sps.uniform.rvs(13,53, size=10)
```

Out[4]:
```
array([ 24.95311267,  34.54291405,  34.64219451,  26.12625998,
        51.85185511,  40.03629702,  13.22918376,  52.60953762,
        63.0083822 ,  17.09470067])
```

Сгенерируем выборки разного размера и посчитаем статистику на ней

In [7]:
```
sample_1 = pd.Series(sps.uniform.rvs(13,53,size=1e7))
sample_1.describe()
```

```
/usr/local/lib/python3.5/site-packages/scipy/stats/_continuous_dis
tns.py:4894: VisibleDeprecationWarning: using a non-integer number
instead of an integer will result in an error in the future
  return self._random_state.uniform(0.0, 1.0, self._size)
```

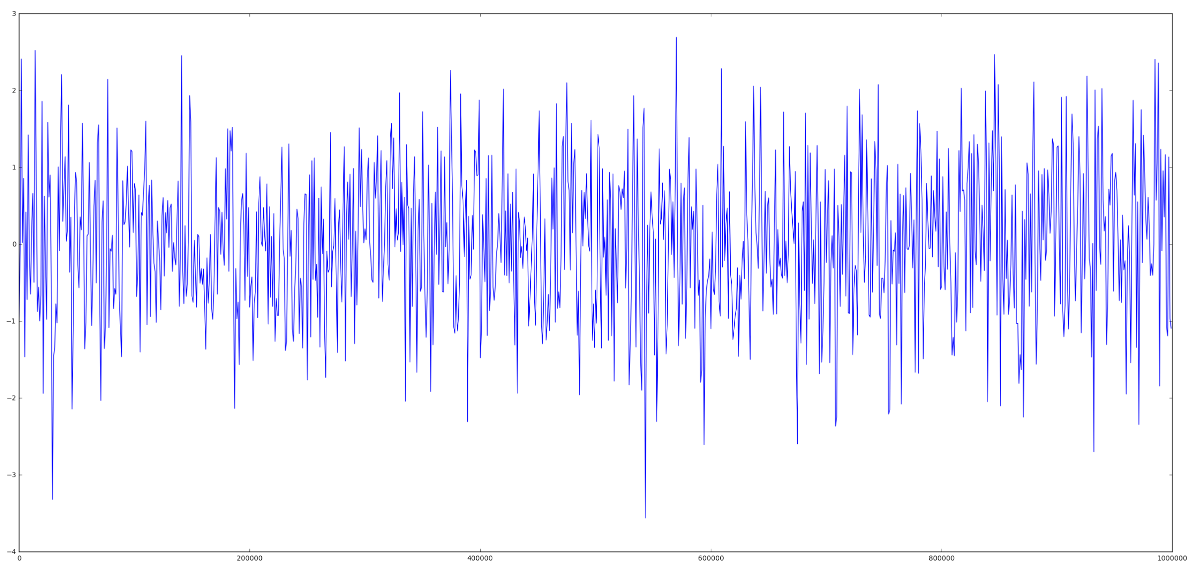
Out[7]:
```
count    1.000000e+07
mean     3.949649e+01
std      1.530035e+01
min      1.300003e+01
25%      2.625041e+01
50%      3.950014e+01
75%      5.273629e+01
max      6.599997e+01
dtype: float64
```

In [7]:
```
results = []
for n in tqdm(range(100, int(1e6), 1000)):
    sample = sps.uniform.rvs(13,53, size=n)
    T_n = (sample.sum() - n * matozh) / np.sqrt(n * variation)
    results.append(T_n)
```

```
100%|██████████| 1000/1000 [00:16<00:00, 62.18it/s]
```

In [9]:
```
plt.figure(figsize=(30,14))
plt.plot(range(100, int(1e6), 1000), results)
plt.show()
```

```
<matplotlib.figure.Figure at 0x107a7fb38>
```



Теперь возьмем довольно большую выборку и посчитаем статистику на ее префиксах(по всем k <= n)

In [28]:
```
# COlor generator
from color_generator import ColorGenerator
```

In [25]:
```
col_gen = ColorGenerator()
```

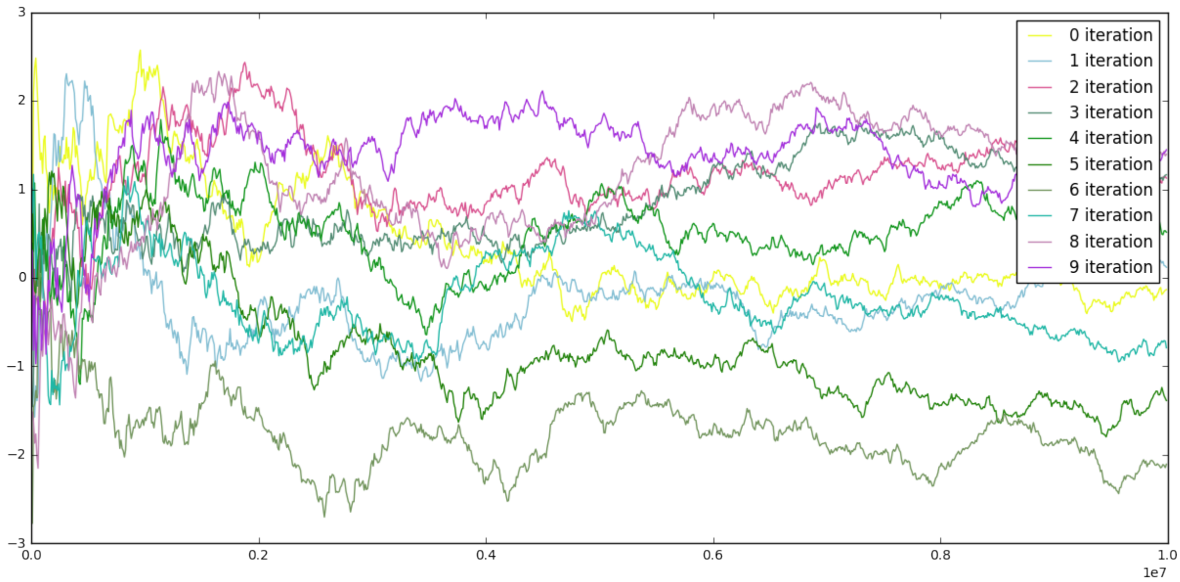
In [27]:
```
col_gen.get_color()
```

Out[27]: '#3198E5'

In [31]:
```
def draw_stats(size=int(1e7), step=int(1e4), times=1):
    all_res = []
    plt.figure(figsize=(15,7))
    for tim in tqdm(range(times)):
        sample = sps.uniform.rvs(13,53, size=size)
        cumsums = np.cumsum(sample)
        results = []
        for k in tqdm(range(0, size, step)):
            T_k = (cumsums[k] - k * matozh) / np.sqrt(k * variation
)
            results.append(T_k)
        all_res.append(results)
        plt.plot(range(0, size, step), results, color=col_gen.get_c
olor(), label=str(tim) + ' iteration')
    plt.legend()
    plt.show()
    return all_res
```

In [33]:
```
results =  draw_stats(times=10)
```

```
  0%|          | 0/10 [00:00<?, ?it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
 10%|█         | 1/10 [00:00<00:02,  3.13it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
 20%|██        | 2/10 [00:00<00:02,  3.02it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
 30%|███       | 3/10 [00:00<00:02,  3.14it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
 40%|████      | 4/10 [00:01<00:01,  3.25it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
 50%|█████     | 5/10 [00:01<00:01,  3.30it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
 60%|██████    | 6/10 [00:01<00:01,  3.39it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
 70%|███████   | 7/10 [00:02<00:00,  3.44it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
 80%|████████  | 8/10 [00:02<00:00,  3.48it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
 90%|█████████ | 9/10 [00:02<00:00,  2.67it/s]
  0%|          | 0/1000 [00:00<?, ?it/s][A
100%|██████████| 10/10 [00:03<00:00,  2.75it/s]
```



При росте числа элементов скачкИ становятся меньше. Ответ на вопрос про сходимость для произвольной реализации. Да, может сходиться к нулю, например в случае, когда все элементы выборки будут равны Матожиданию, тогда $\forall n : T_n = 0$

ЦПТ выполняется, т.к. условие теоремы выполнено. Нам дан набор н.о.р.с.в. с конечным вторым моментом, ну тогда выполнена ЦПТ для T_n - ведь это она и будет стремиться к стандартной нормальной.

In [44]:
```
len(results[0])
```

Out[44]: 1000

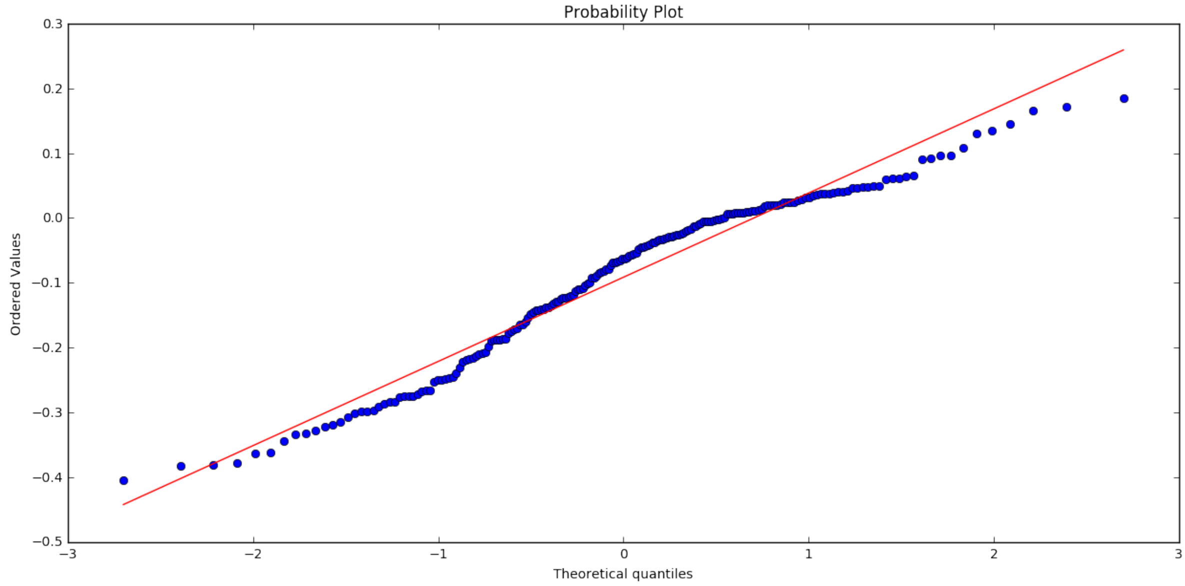
Можем так же проверить нулевую гипотезу о нормальности с помощью теста Шапиро-Уилка. Смотря на последние 110 ответов из полученного массива, нельзя отвергнуть гипотезу на уровне доверия 0.05.

In [52]:
```
sps.shapiro(results[0][890:])
```

Out[52]:
```
(0.9790902137756348, 0.08141706883907318)
```

Можем для наглядности построить еще Q-Q plot.

In [58]:
```
plt.figure(figsize=(15, 7))
sps.probplot(results[0][800:], plot=plt)
plt.show()
```



Супер! не сильно отличается от нормального (: