In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [ ]:
```
algorithms = [
    "naive",
    "kmp-standard",
    "kmp-refined",
    "z-function",
]

texts = []
for i in range(0, 5):
    texts += [
        f"wildcard-{i}-small-bin",
        f"wildcard-{i}-small-sq",
        f"wildcard-{i}-big-bin",
        f"wildcard-{i}-big-sq",
    ]
```

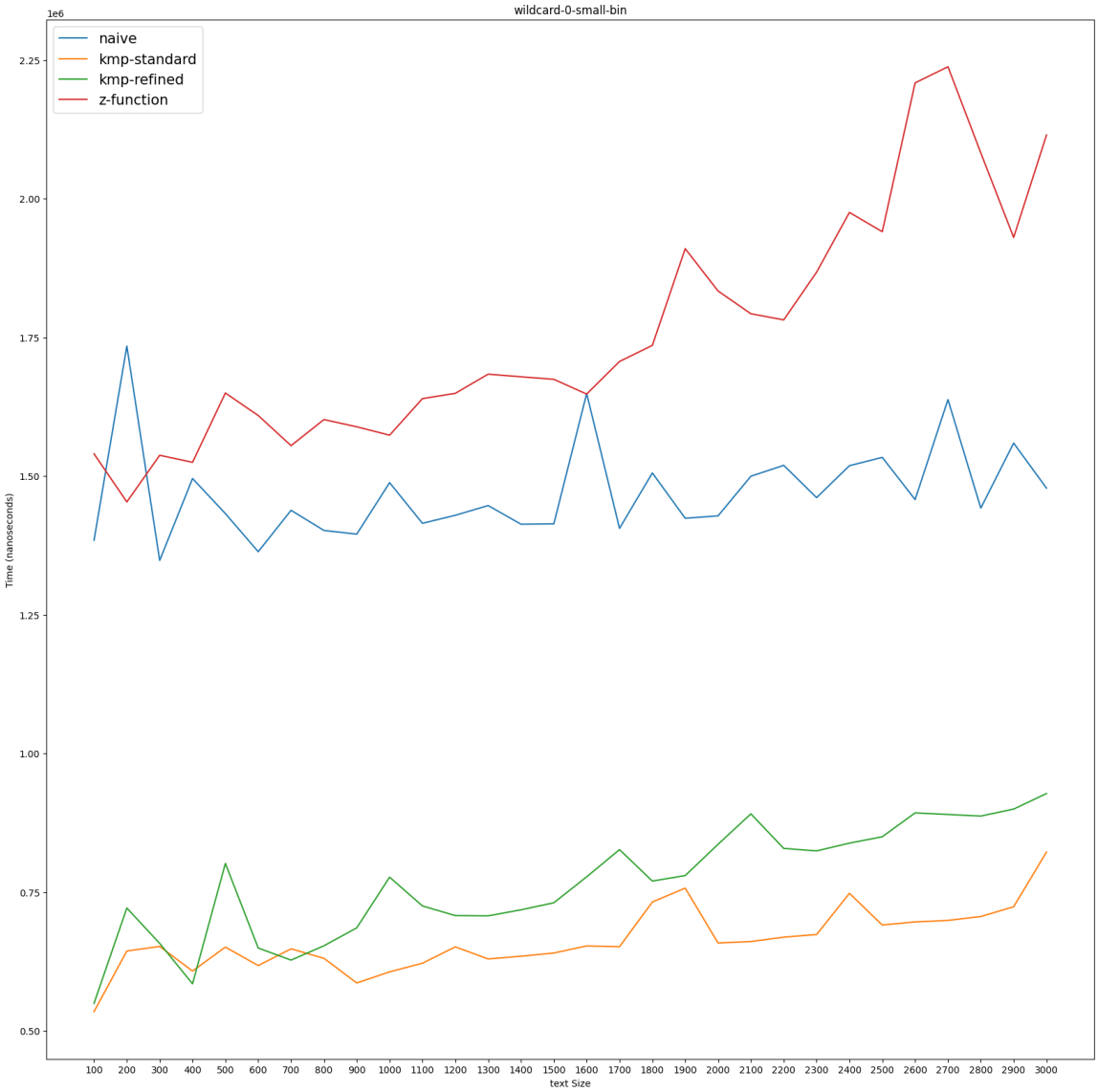
In [ ]:
```
data = pd.read_csv('../data.csv', sep=';', header=None)
data.columns = ['algorithm', 'text', 'size', 'time']
```

In [ ]:
```
def print_text(data, text):
    text_df = data[data['text'] == text]
    plt.figure(figsize=(20, 20))
    for algorithm in algorithms:
        df = text_df[text_df['algorithm'] == algorithm]
        plt.plot(df['size'], df['time'], label=algorithm)
    plt.title(text)
    plt.xlabel('text Size')
    plt.xticks(text_df['size'].unique())
    plt.ylabel('Time (nanoseconds)')
    plt.legend(labelcolor='black', prop={'size': 15})
```

Зависимость времени выполнения от размера паттерна

Без символов подстановки, 10000 символов, бинарный алфавит

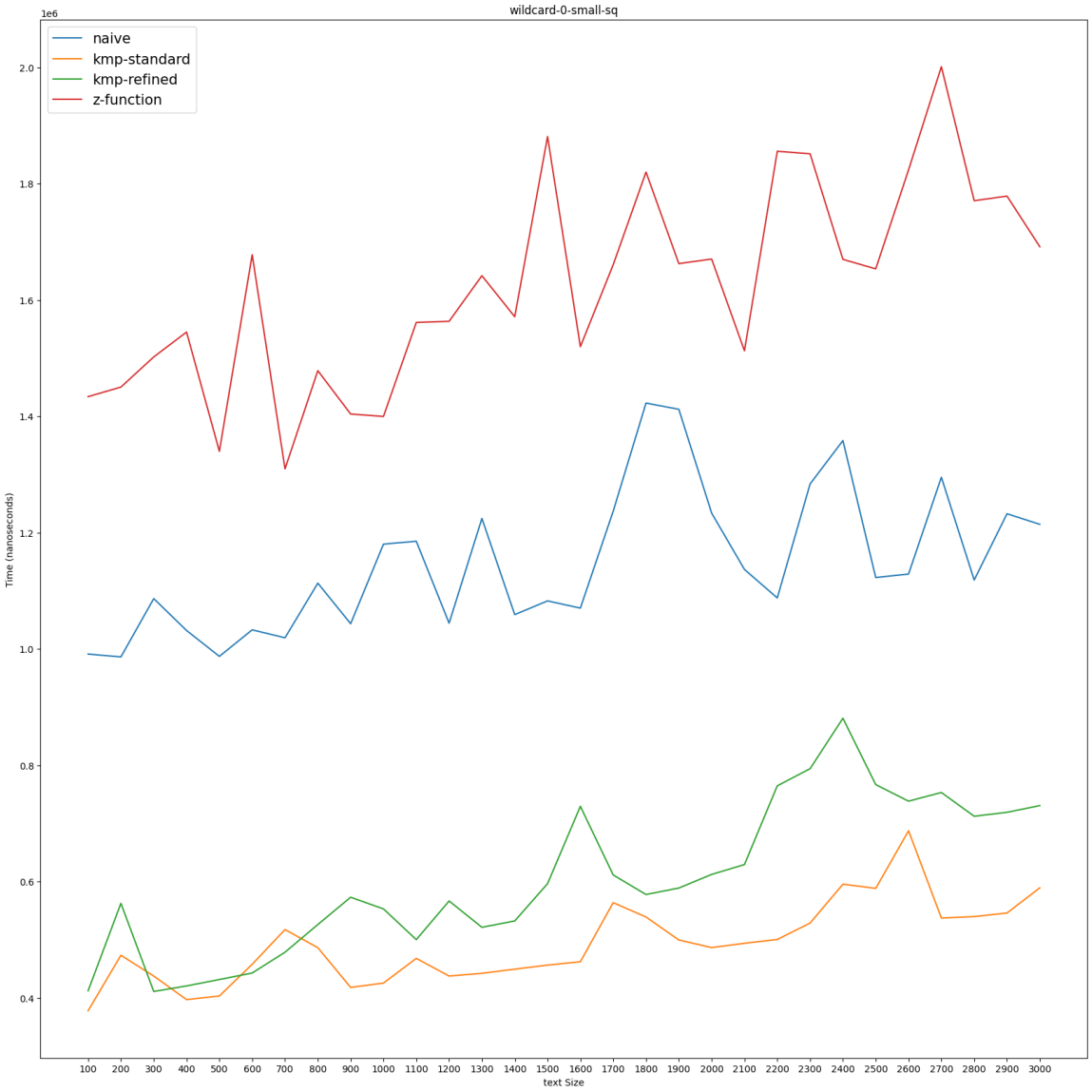
In [ ]:
```
print_text(data, texts[0])
```



Вывод: Алгоритм КМП и его оптимизация уточнёнными гранями показали себя ожидаемо лучше наивного алгоритма. У Z-функции явно большая константа и, возможно, проблема с реализацией, поэтому она отработала хуже квадратичного алгоритма

2. Без символов подстановки, 10000 символов, 4-символьный алфавит

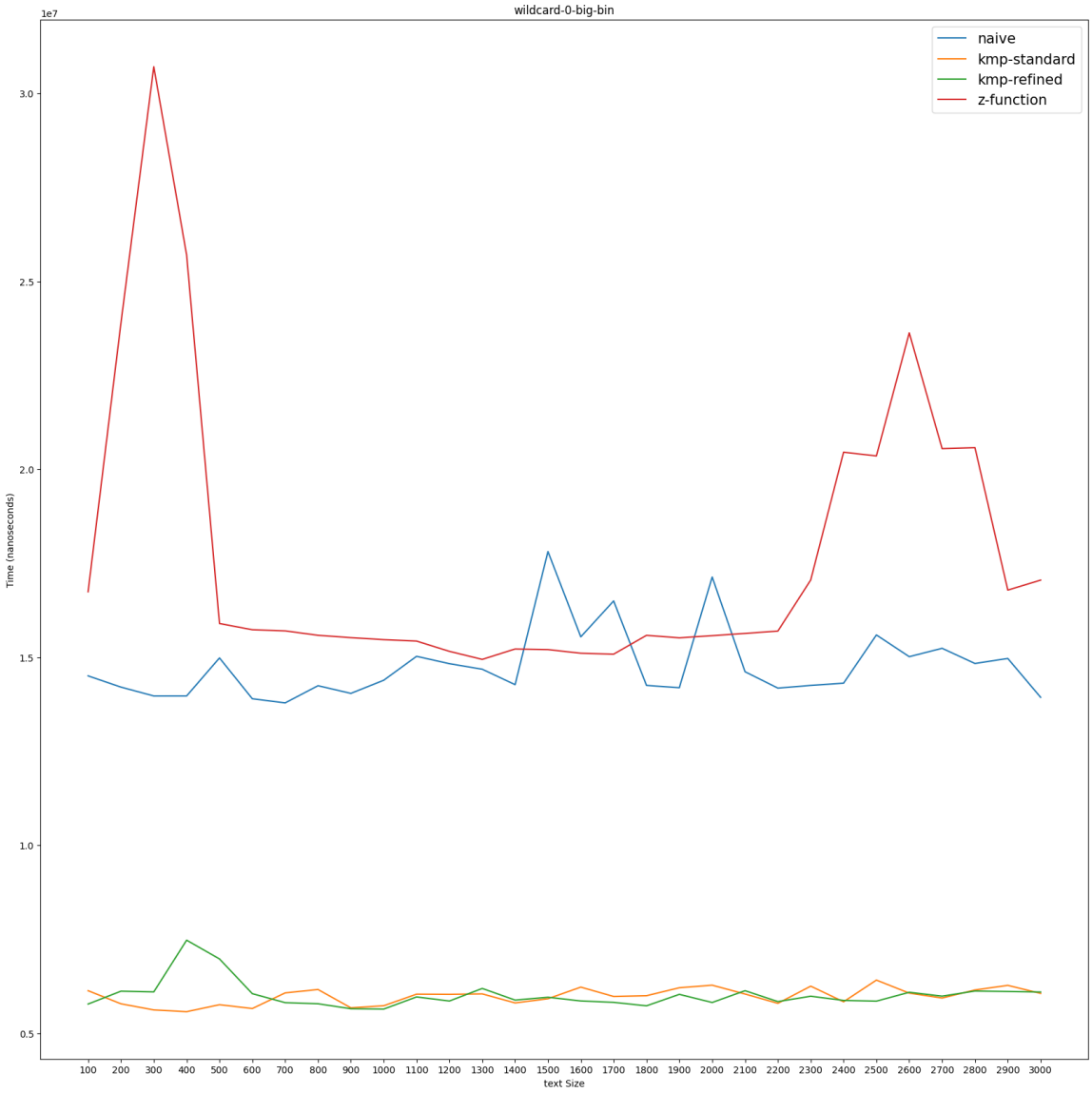
In [ ]:
```
print_text(data, texts[1])
```



Вывод: Аналогично предыдущему пункту реализации КМП ведут себя ожидаемо, наивный алгоритм, в целом, тоже, а вот Z-функция показала себя плохо. Идейно уточнённый КМП должен быть лучше неуточнённого, но фактически не так

3. Без символов подстановки, 100000 символов, бинарный алфавит

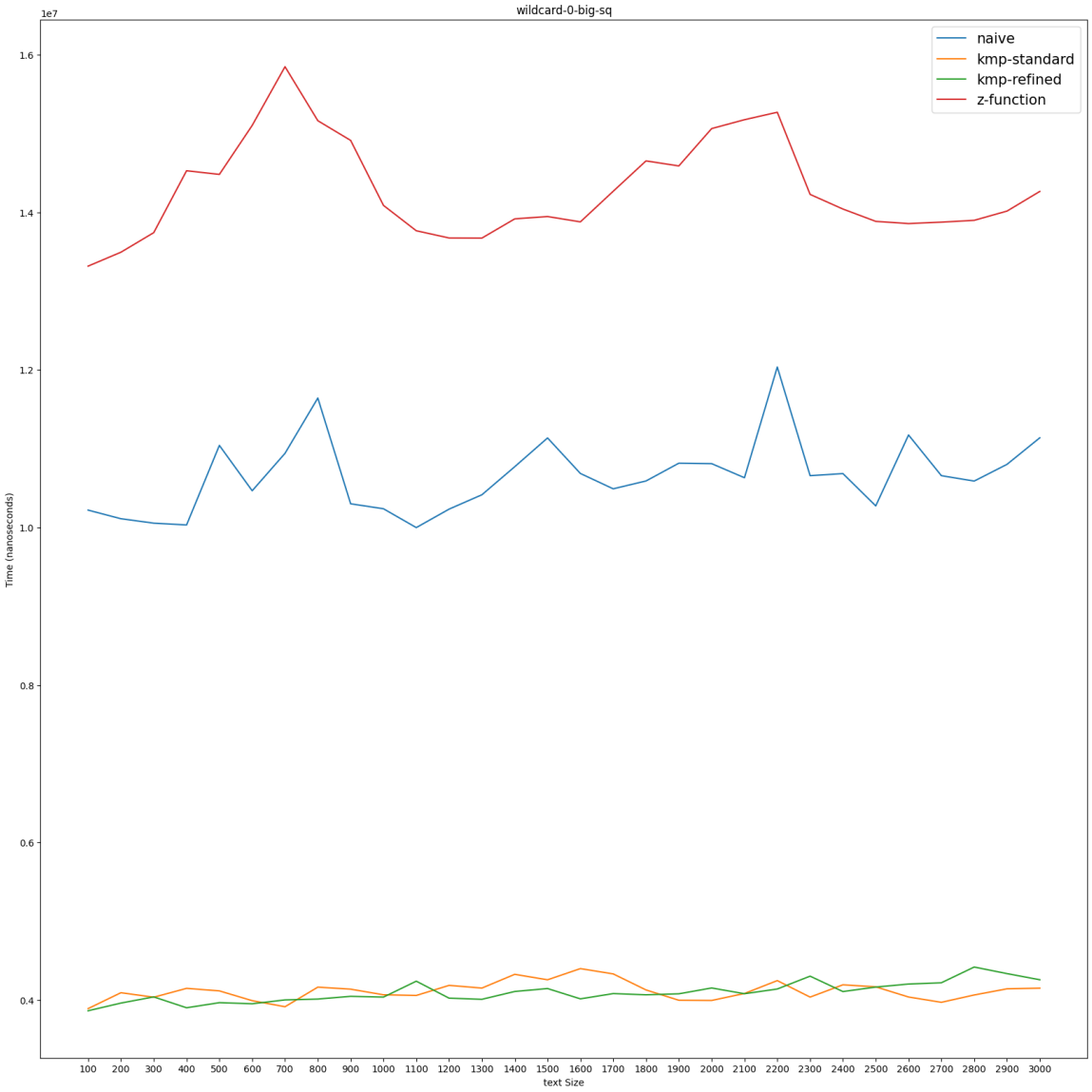
In [ ]:
```
print_text(data, texts[2])
```



Вывод: Тут явно видно преимущество КМП, но наивный алгоритм почему-то не уходит в квадрат

4. Без символов подстановки, 100000 символов, 4-символьный алфавит

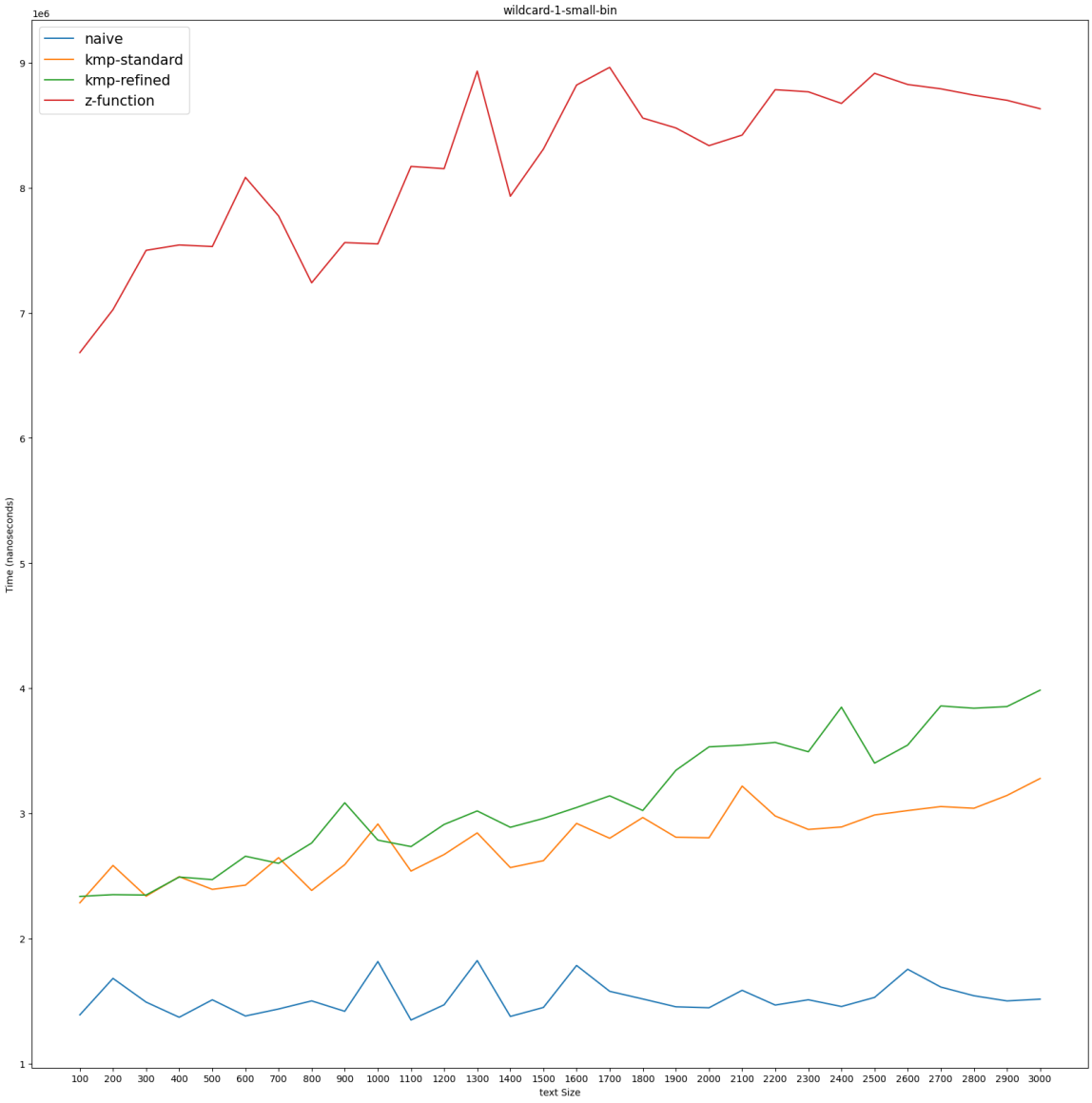
In [ ]:
```
print_text(data, texts[3])
```

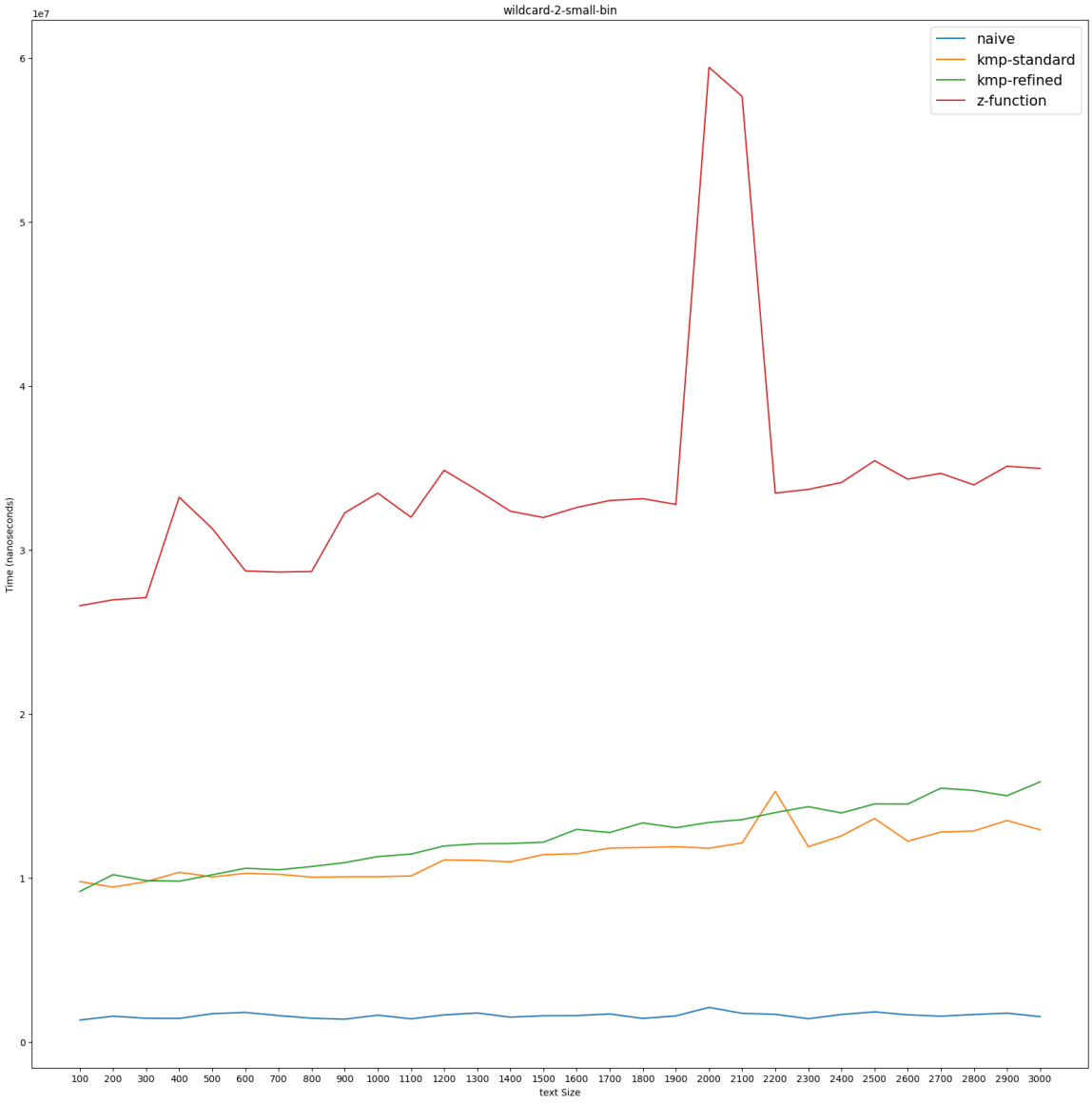


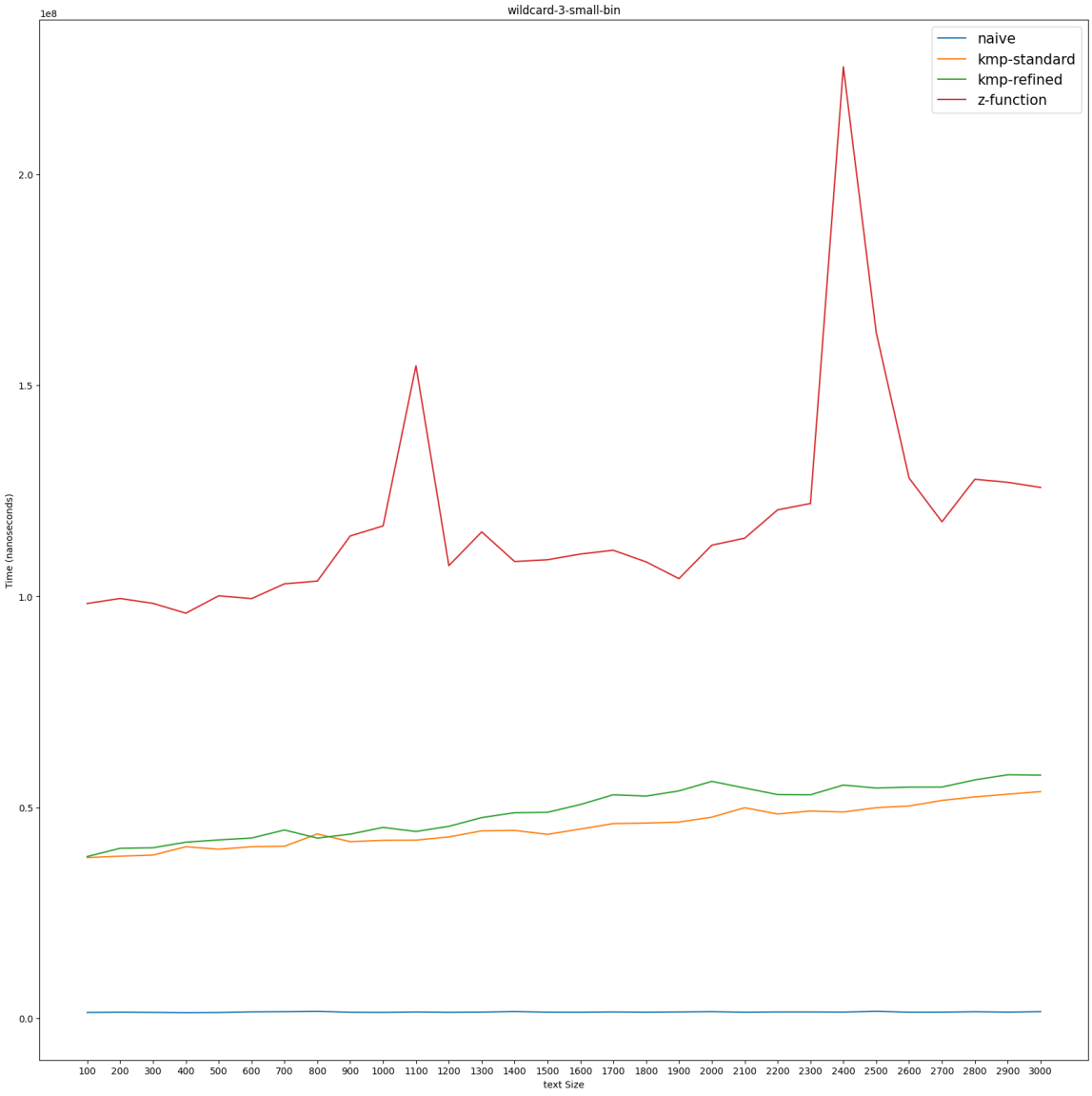
Вывод: Аналогично: КМП заметно лучше наивного алгоритма, а Z-функция показала себя плохо

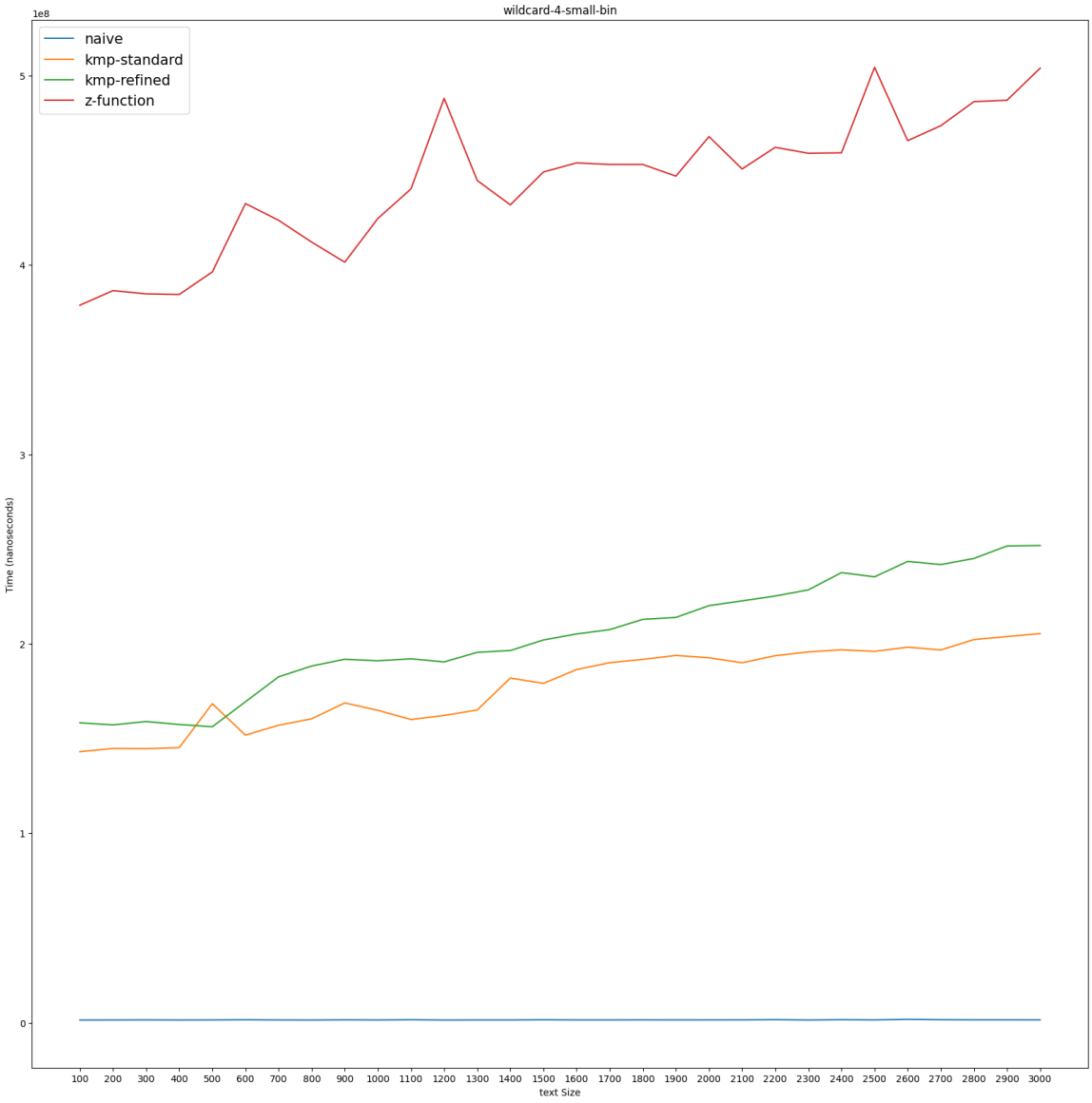
5. С 1-4 символами подстановки, 10000 символов, бинарный алфавит

In [ ]:
```
print_text(data, texts[4])
print_text(data, texts[8])
print_text(data, texts[12])
print_text(data, texts[16])
```


wildcard-1-small-bin
1e6
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size


wildcard-2-small-bin
1e7
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
0
1
2
3
4
5
6
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000

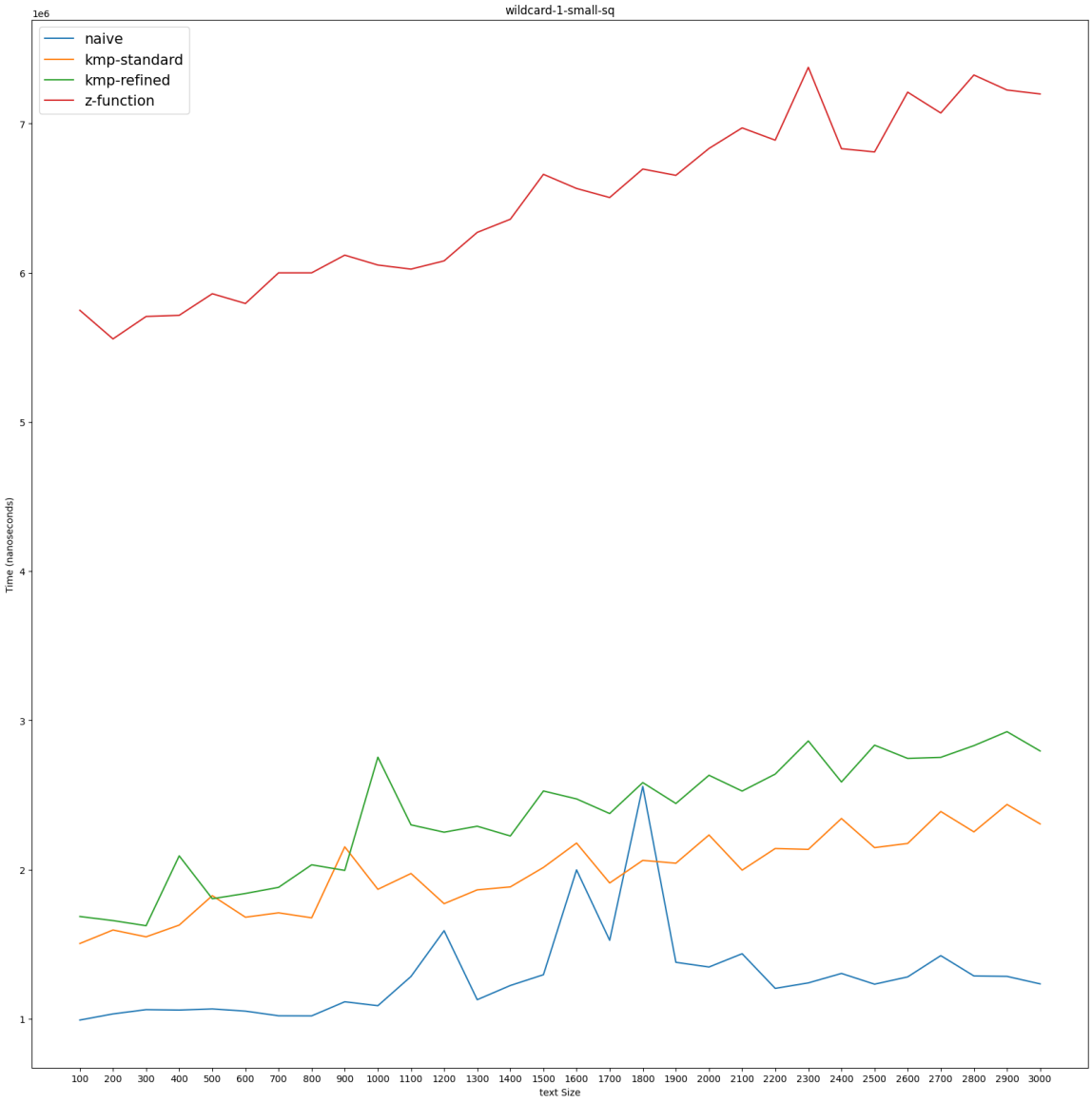

wildcard-3-small-bin
1e8
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
0.0
0.5
1.0
1.5
2.0
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000

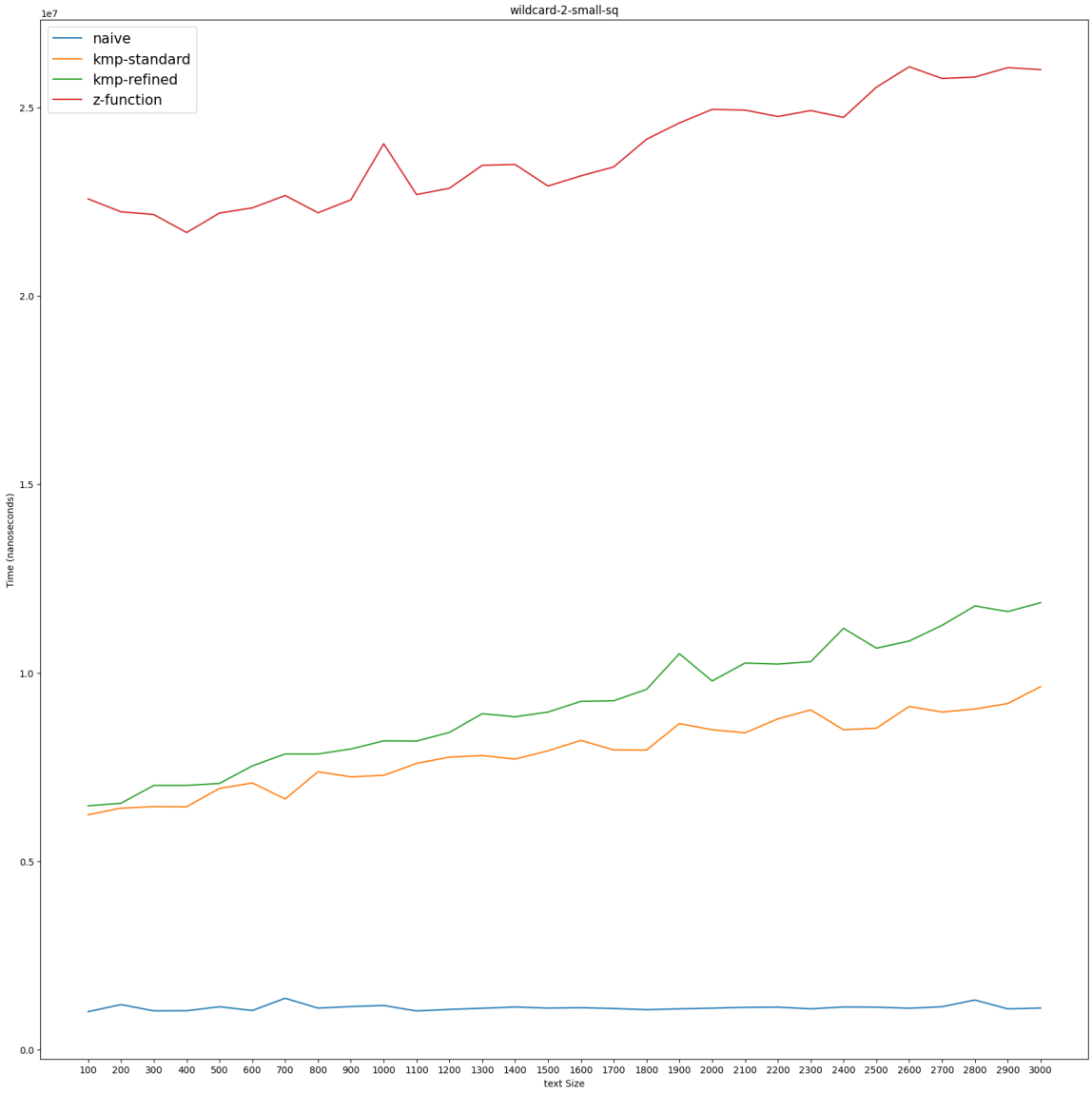


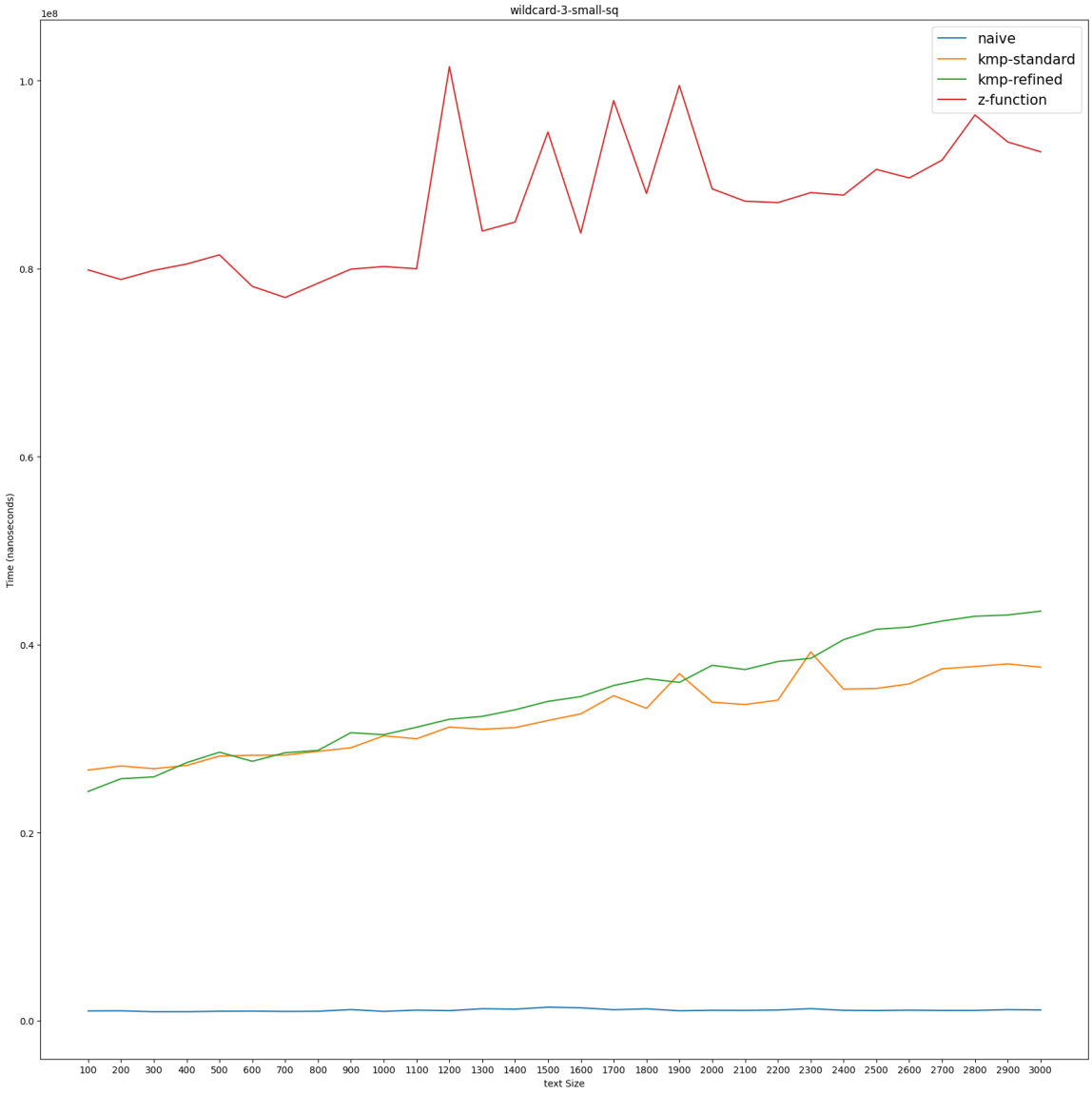
Вывод: Здесь очевидна проблема из-за константы у всех алгоритмов, кроме наивного, потому что используется перебор по 4^4=256 строкам

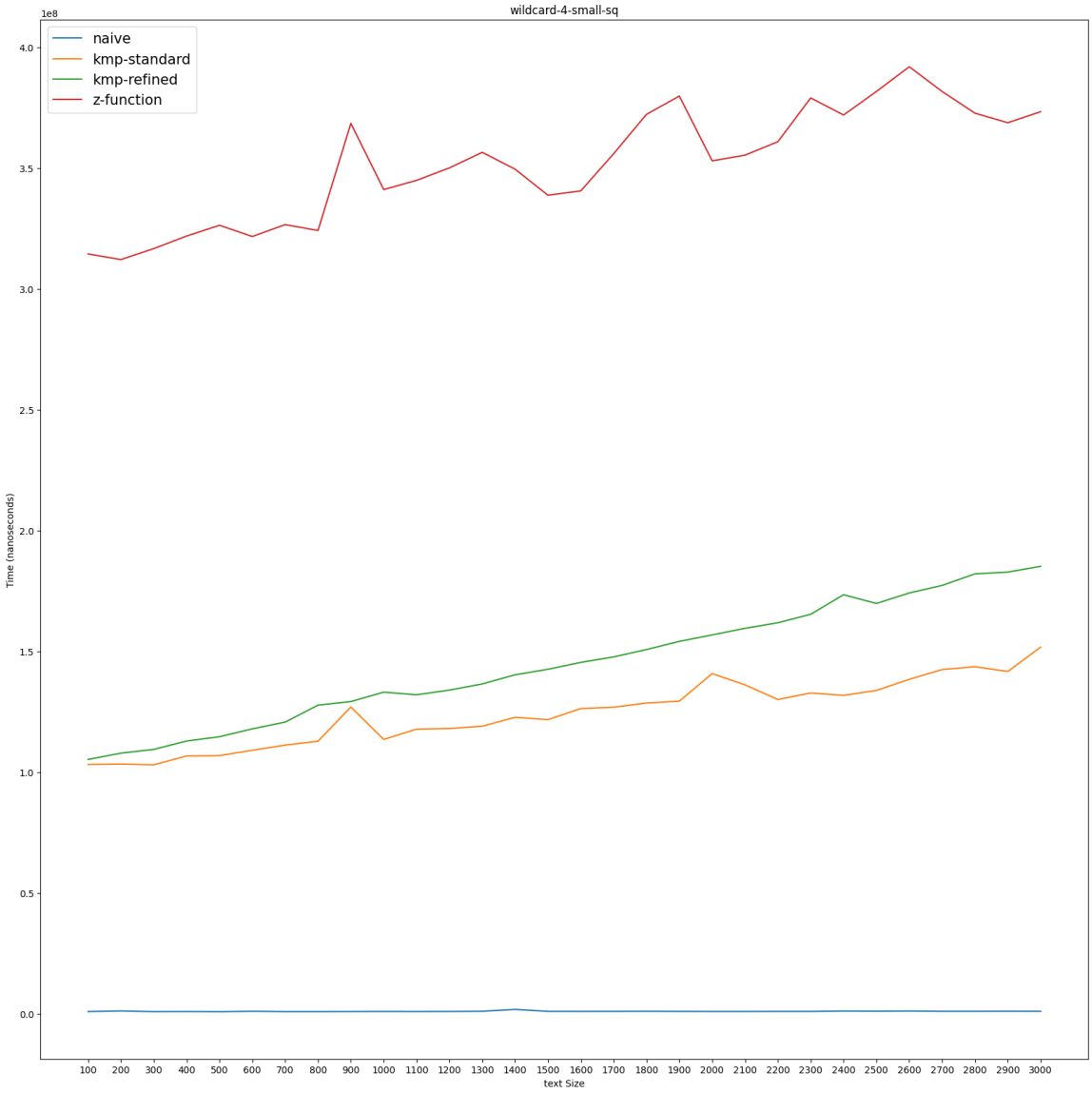
6. С 1-4 символами подстановки, 10000 символов, 4-символьный алфавит

In [ ]:
```
print_text(data, texts[5])
print_text(data, texts[9])
print_text(data, texts[13])
print_text(data, texts[17])
```


wildcard-1-small-sq
1e6
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
1
2
3
4
5
6
7
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000


wildcard-2-small-sq
1e7
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
0.0
0.5
1.0
1.5
2.0
2.5
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000

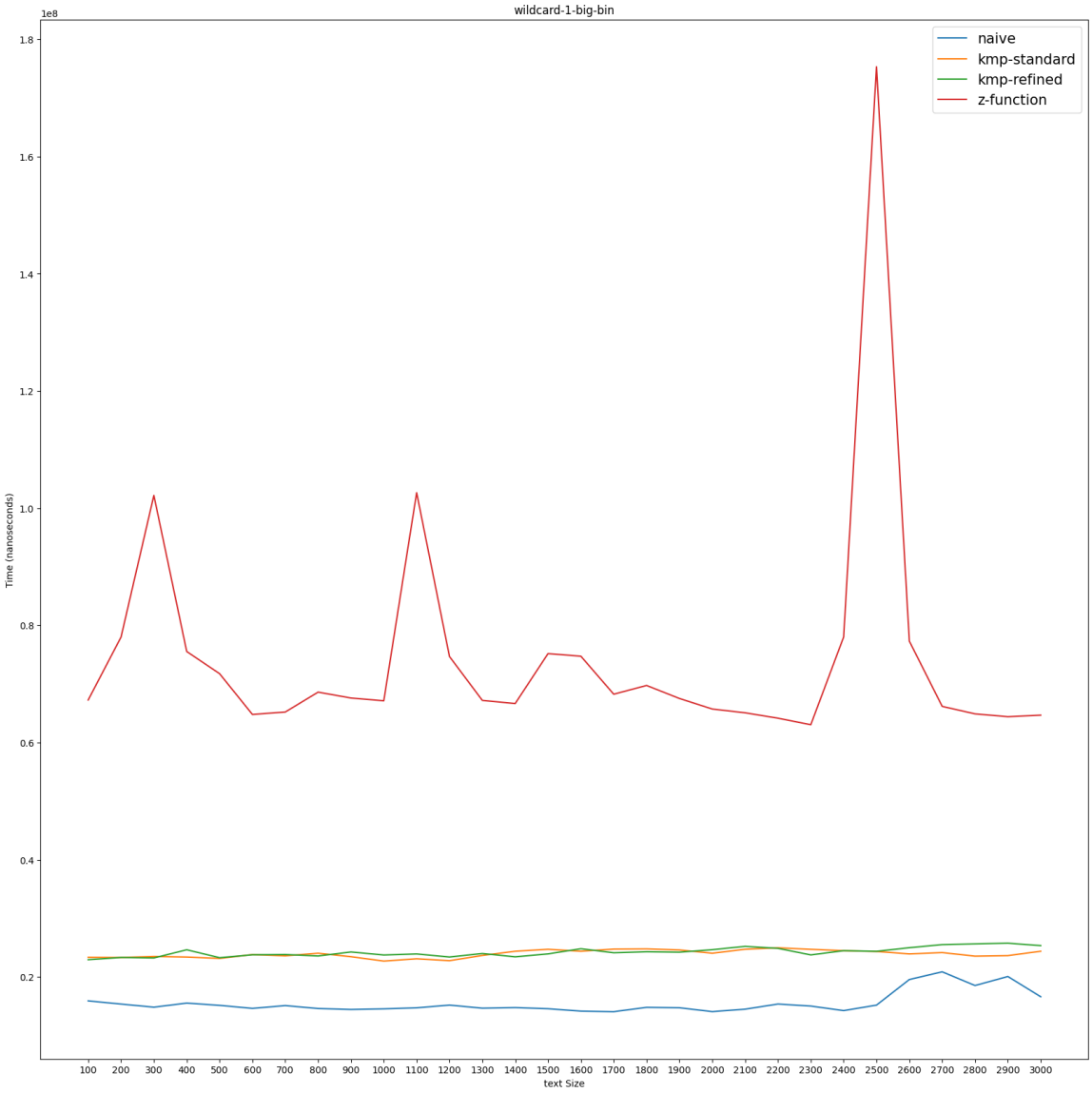

wildcard-3-small-sq
1e8
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
0.0
0.2
0.4
0.6
0.8
1.0
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000

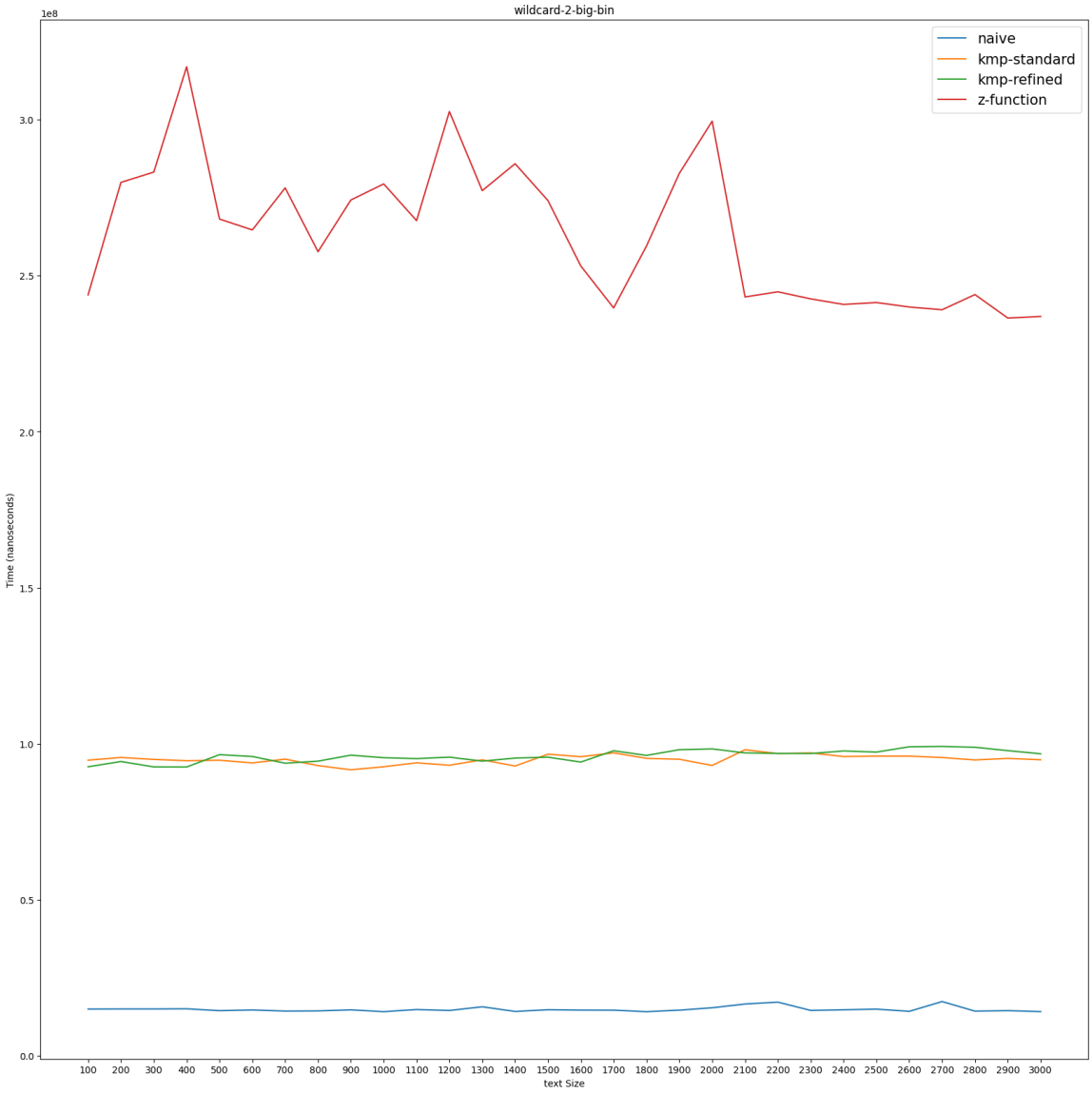


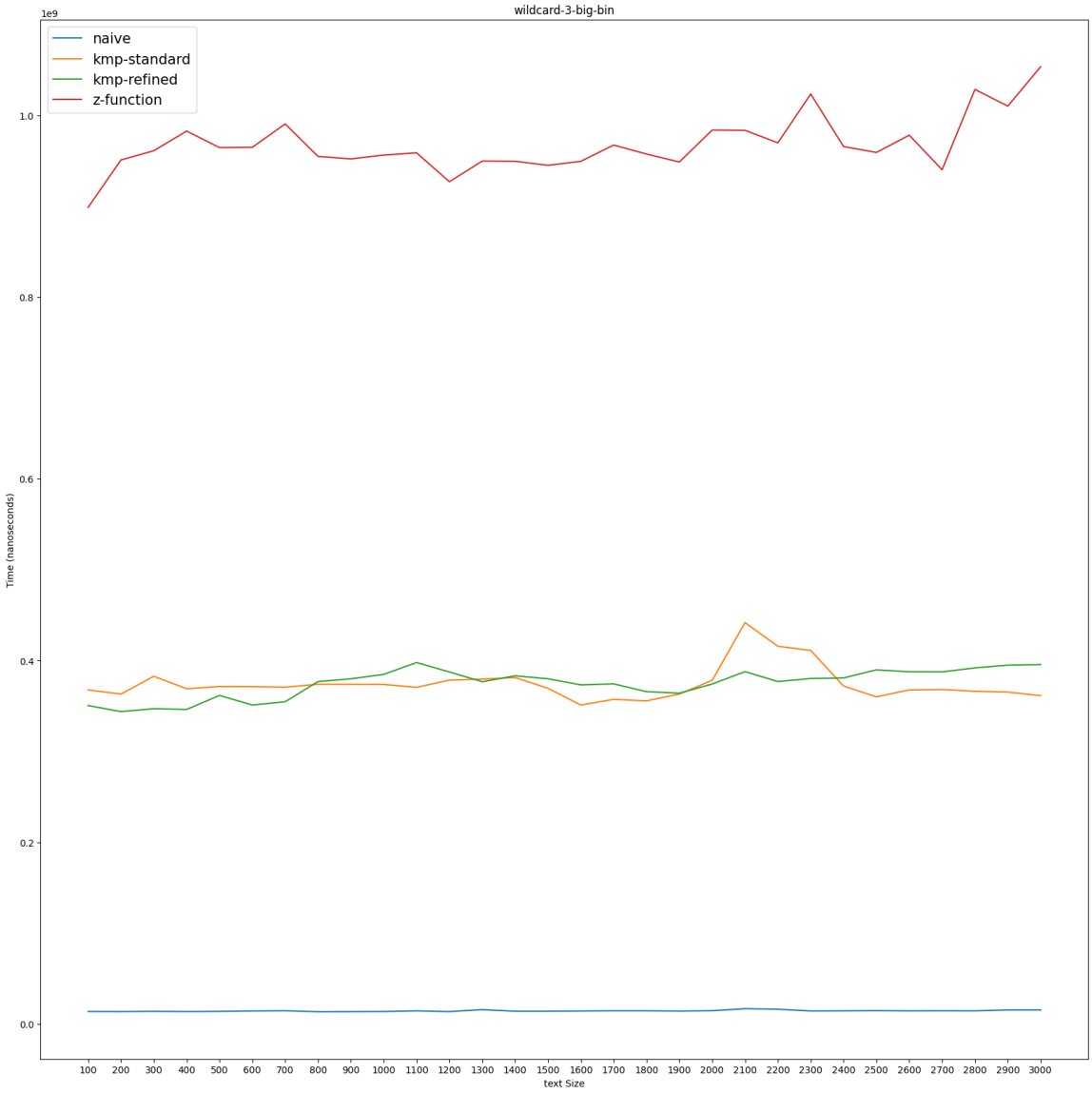
Вывод: Из-за способа обработки символа подстановки нельзя сказать ничего содержательного по результатам измерений

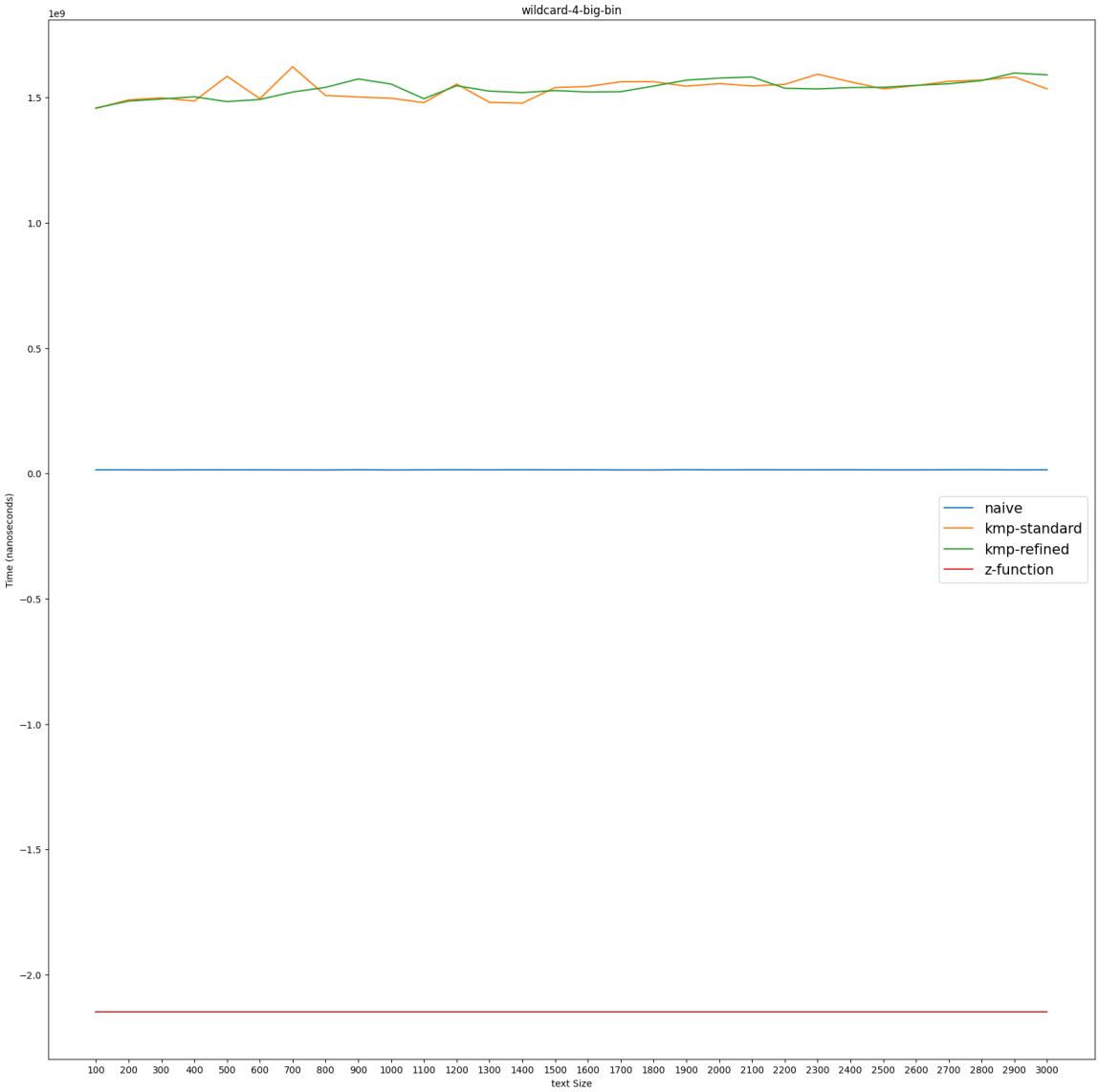
7. С 1-4 символами подстановки, 100000 символов, бинарный алфавит

In [ ]:
```
print_text(data, texts[6])
print_text(data, texts[10])
print_text(data, texts[14])
print_text(data, texts[18])
```


wildcard-1-big-bin
1e8
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000


wildcard-2-big-bin
1e8
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
0.0
0.5
1.0
1.5
2.0
2.5
3.0
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000

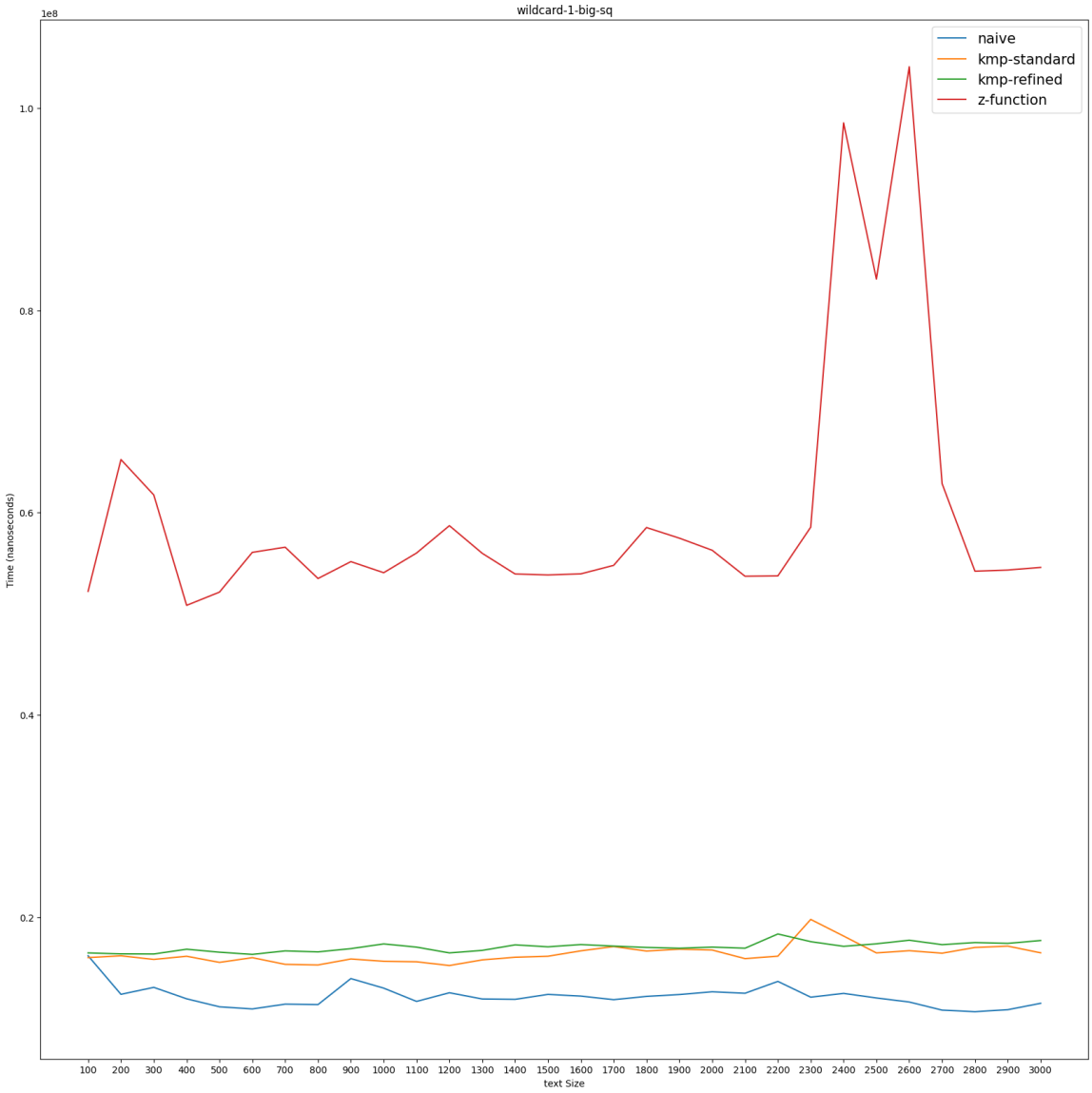

wildcard-3-big-bin
1e9
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
0.0
0.2
0.4
0.6
0.8
1.0
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000

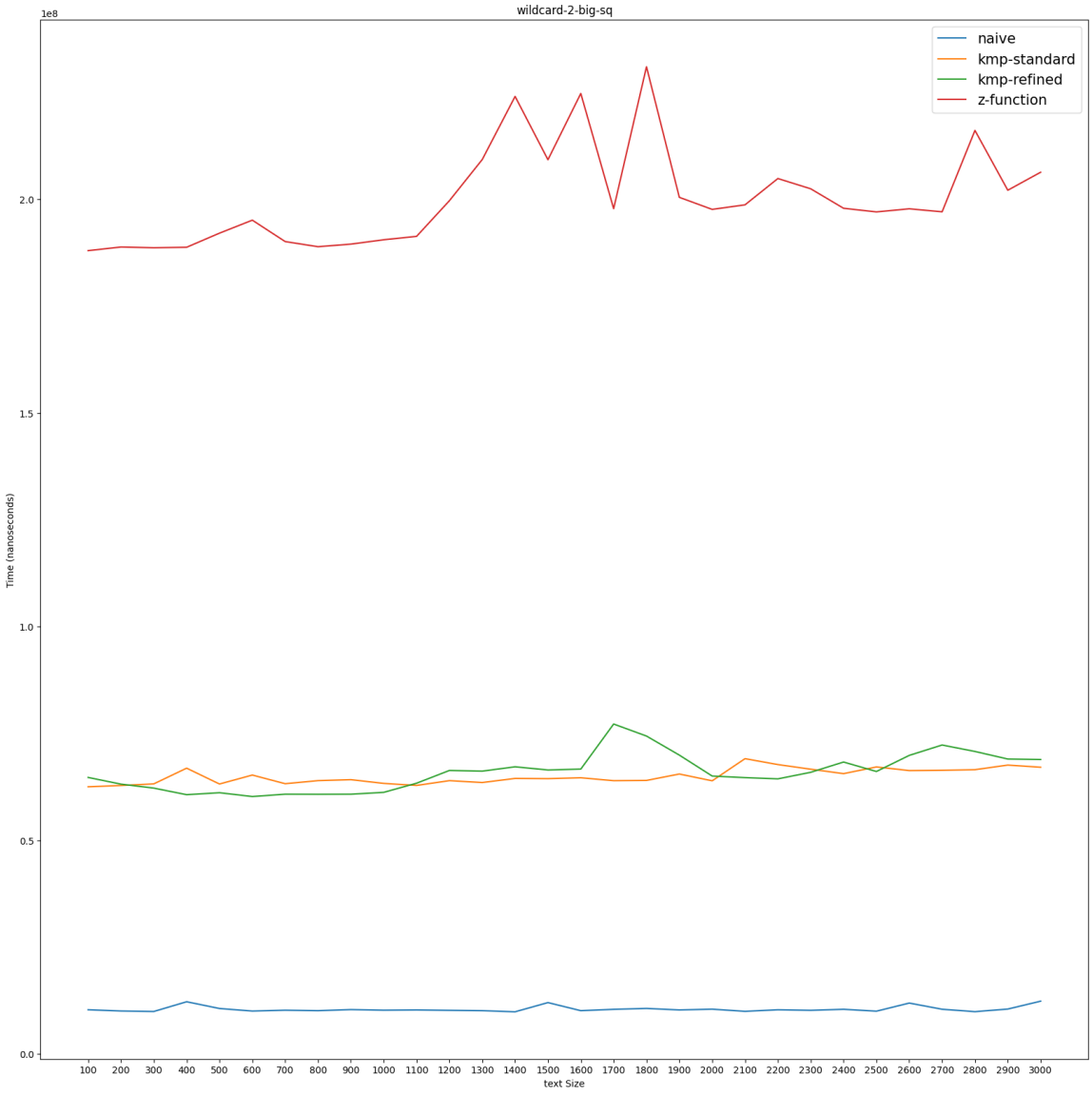


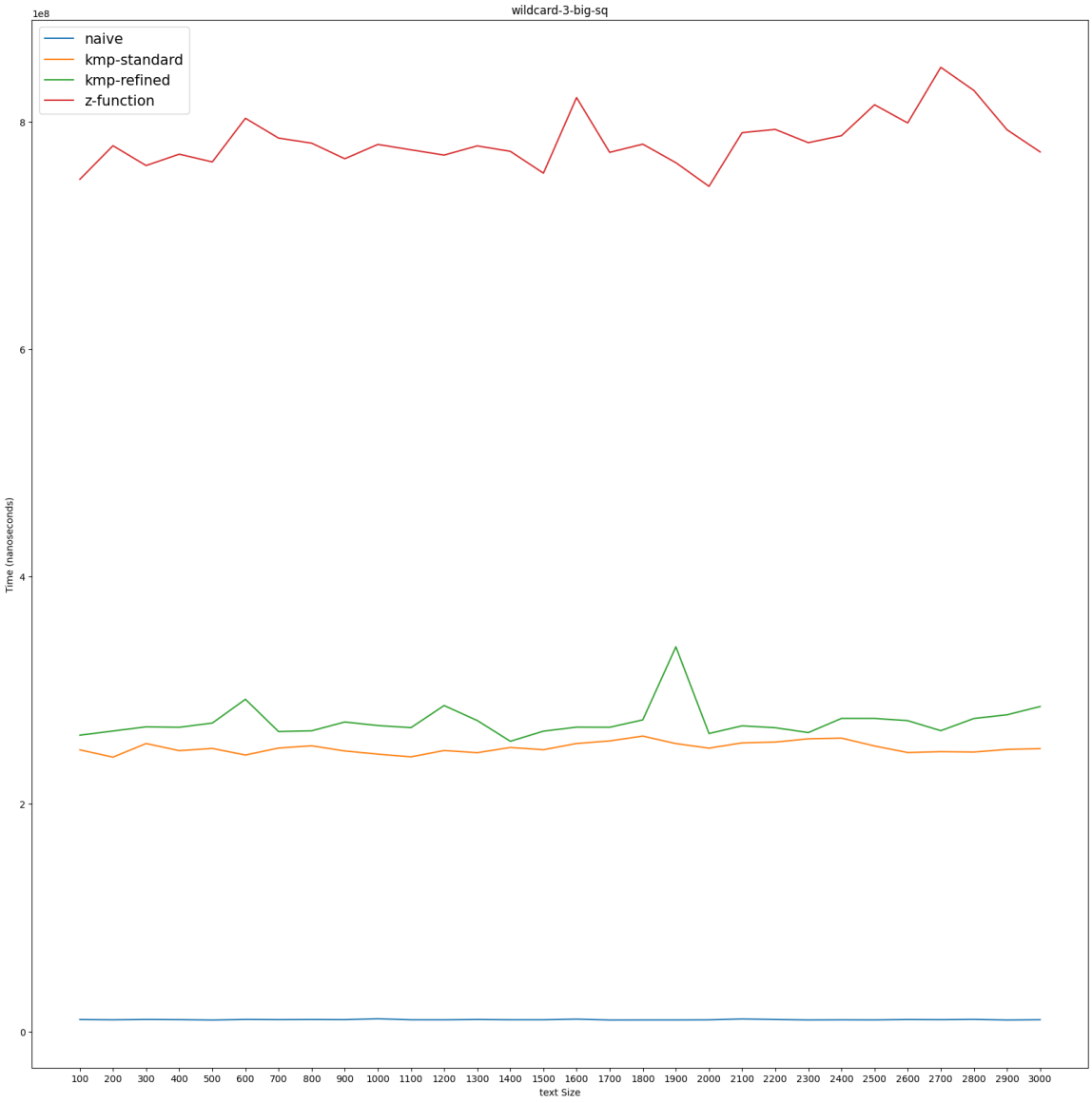
Вывод: Увеличение длины исходного текста сделало проблему только нагляднее, а в последнем измерении время исполнения Z-функции даже переполнилось

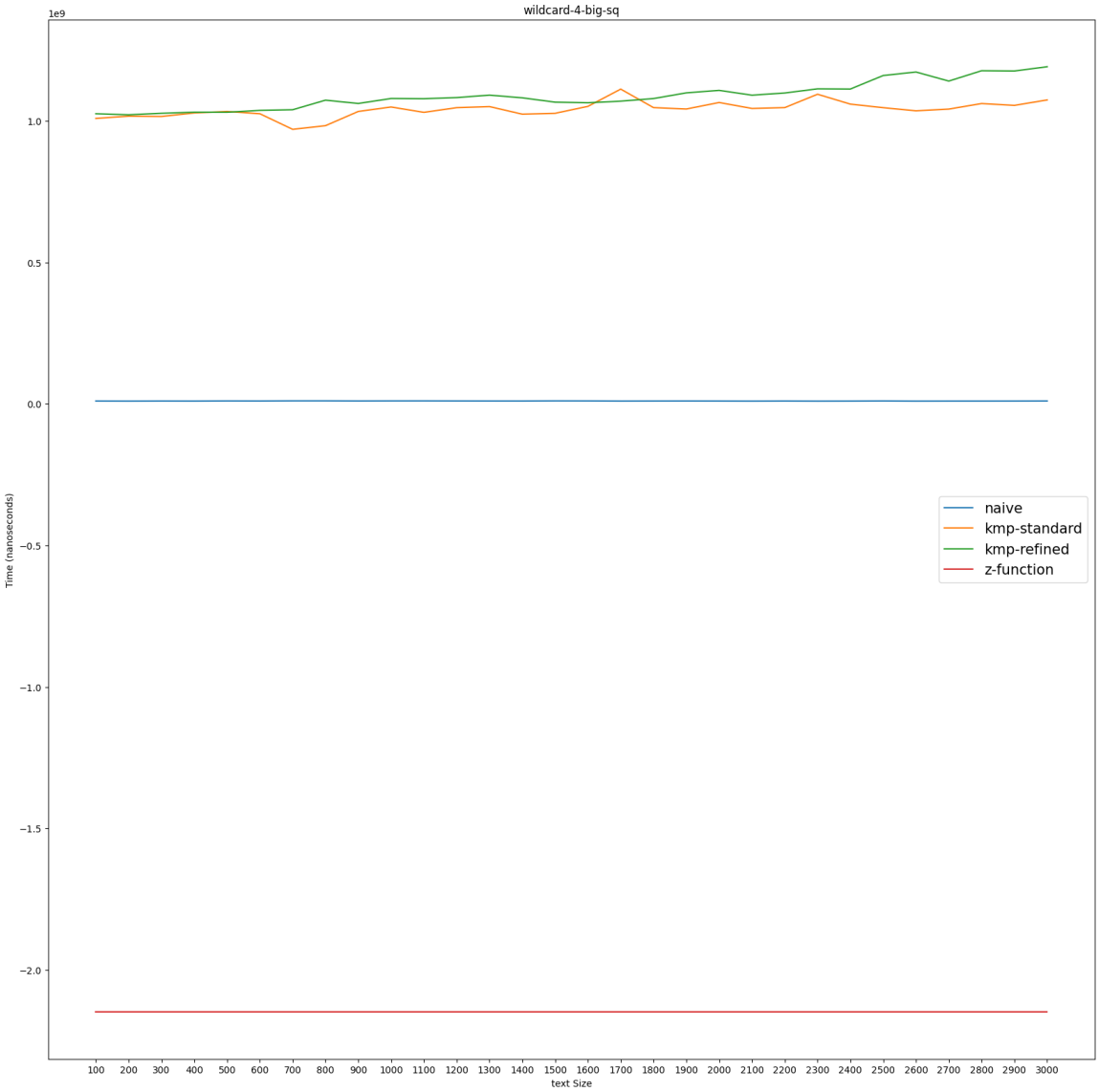
8. С 1-4 символами подстановки, 100000 символов, 4-символьный алфавит

In [ ]:
```
print_text(data, texts[7])
print_text(data, texts[11])
print_text(data, texts[15])
print_text(data, texts[19])
```


wildcard-1-big-sq
1e8
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
1.0
0.8
0.6
0.4
0.2
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000


wildcard-2-big-sq
1e8
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
text Size
0.0
0.5
1.0
1.5
2.0
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000


wildcard-3-big-sq
1e8
naive
kmp-standard
kmp-refined
z-function
Time (nanoseconds)
0
2
4
6
8
100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000
text Size



Вывод: Идентично предыдущему пункту...