Laboratorium Bezpieczeństwa Systemów Teleinformatycznych (cz. 2 – Testy DieHard)

Data oddania: 31.05.2021

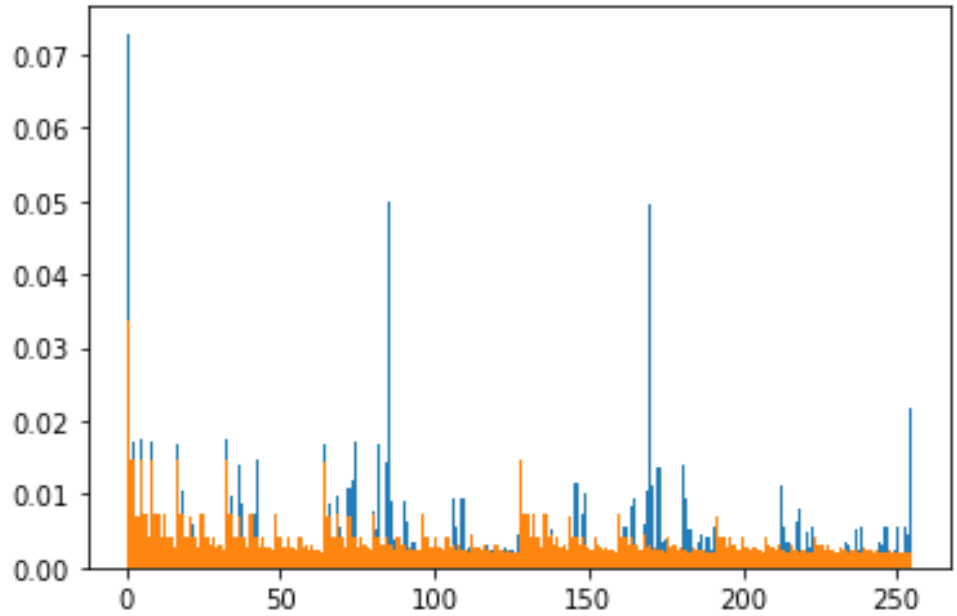
Wykonali: Tomasz Wołoszyn, Patryk Resler

Podstawa opracowania: Liang Zhao, Xiaofeng Liao, Di Xiao, Tao Xiang, Qing Zhou, Shukai Duan. „True random number generation from mobile telephone photo based on chaotic cryptography”

GITHUB: https://github.com/tomasz-woloszyn/Bezpiecze-stwo-System-w-Teleinformatycznych

Zakres danych testowych:

Na potrzeby testów udało się wygenerować 655360 liczb 8 – bitowych, które następnie były przekształcane na potrzeby testów na typ uint32. Rozkład empiryczny poniżej:



Entropia wyliczona zgodnie z $e = -\Sigma_i\, p_i \log_2 (p_i)_i$ dla powyższego rozkładu wynosi: 7,7258 bita.

Test nr 1 – Parking Lot

Na potrzeby przeprowadzenia Parking Lot Test z ciągu danych wyciągnęliśmy wszystkie liczby a następnie, każda liczba 32-bitowa odczytywana jako liczba zmiennoprzecinkowa z zakresu ⟨0,1) rzutowana była na przedział ⟨1,100⟩ wskazujący współrzędne przestrzeni 100×100. W każdym teście wykonano 12 000 prób parkowania zliczając próby zakończone bezkolizyjnie k. Test powtórzono 5-krotnie uzyskując empiryczny rozkład zliczeń prób bezkolizyjnych normalizowany zgodnie ze wzorem testu $(k-3523)/21.9$. W tym teście użyliśmy wygenerowanych 360448 wartości 8 bit, które przekształcone zostały w uint32.

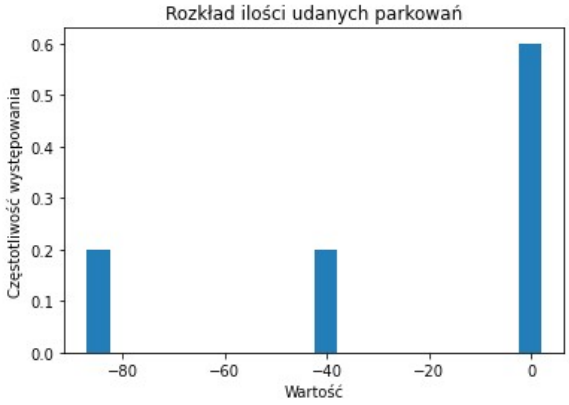
NASZ GENERATOR

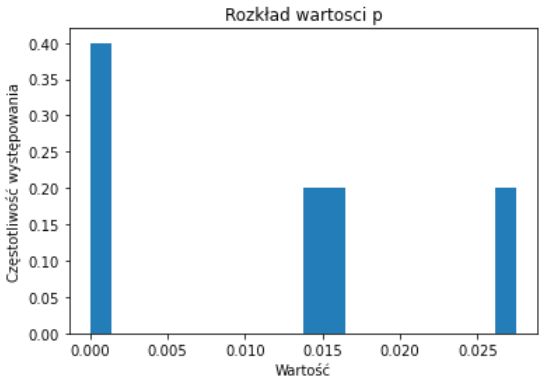
Test wykonany został 5 razy. Jak widać nasze wartości osiągają +/- wartości 3500 czyli maksymalne dopuszczone. Niestety jednak nasz generator nie zdał testu parking Lot, uzyskując wartość p 0.002224682905126391, co nie mieści się w przedziale >0,025 oraz <0,975

```
0 3488
1 3565
2 3482
3 1618
4 2617
```

```
P VALUE:
KstestResult(statistic=0.7449978169635022, pvalue=0.002224682905126391)
```

Poniżej przedstawiłem Rozkład ilości udanych parkowań oraz rozkład wartości p





~ Uwagi

- Mimo małej ilości danych test ten wykonuje się całkiem długo.

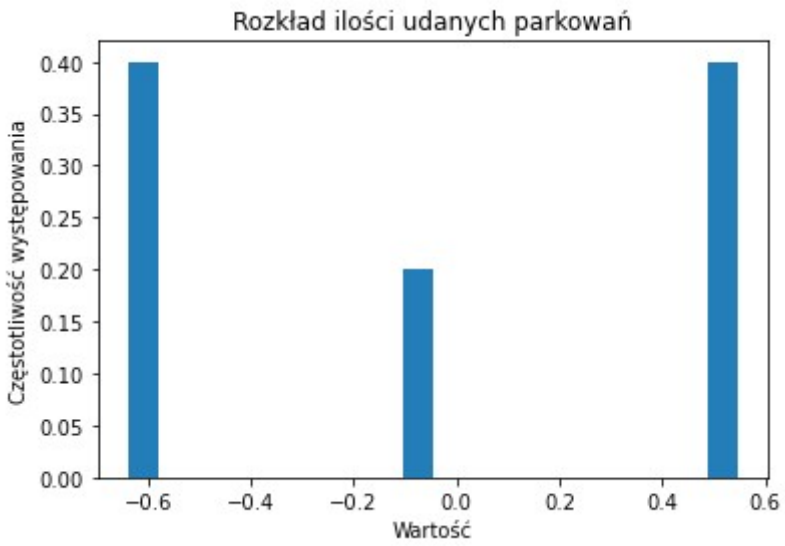
- Liczby losowe z naszego generatora nie spełniły wymagań testu, zatem generator testu nie przeszedł.

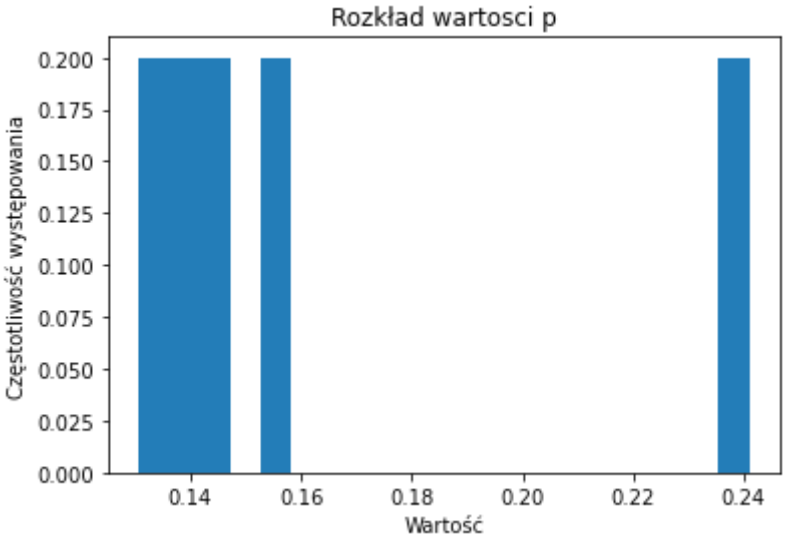
WBUDOWANY GENERATOR RANDOM

Dla wbudowanego generatora każdy test zakończył się pozytywnie. Wartość p value mieści się w przedziale 0,025 do 0,975, zatem test potwierdził losowość wbudowanego generatora liczb losowych.

```
0 3522
1 3535
2 3509
3 3510
4 3534
```

```
P VALUE:
KstestResult(statistic=0.44552081147752864, pvalue=0.20292579791058873)
```


Rozkład ilości udanych parkowań
Częstotliwość występowania
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
Wartość

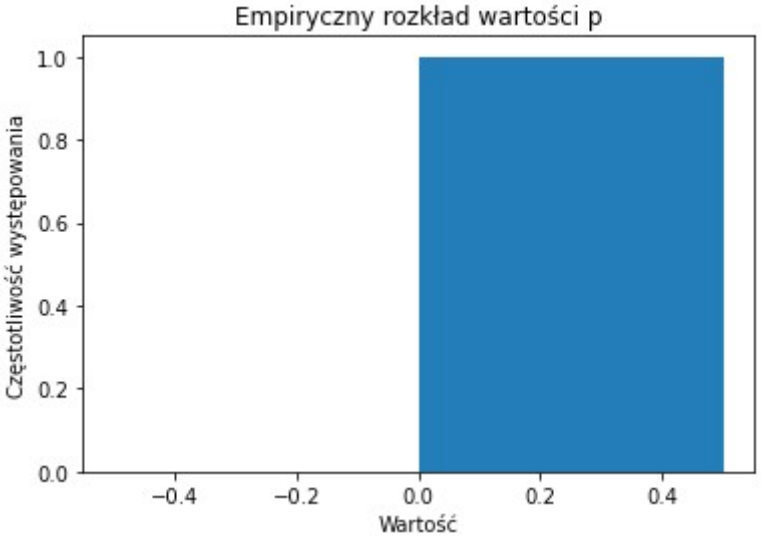

Rozkład wartosci p
Częstotliwość występowania
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0.14
0.16
0.18
0.20
0.22
0.24
Wartość

Test nr 2 – Count the ones

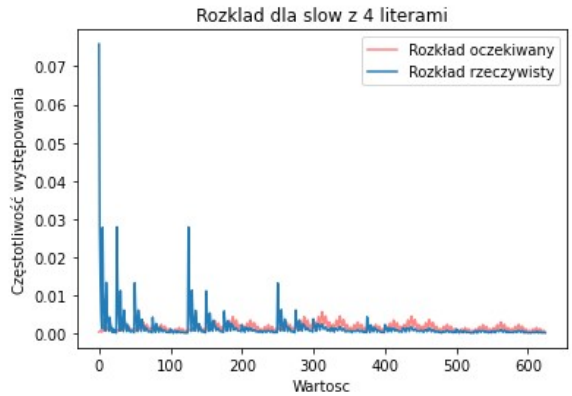
Cały strumień bajtów zostaje podzielony na ciąg nakładających się na siebie 5 literowych słów, gdzie każda litera przyjmuje wartości A,B,C,D,E – jest to zdefiniowane przez liczbę jedynek w bajcie – odpowiednio 0-2:A 3:B 4:C 5:D 6-8:E. Prawdopodobieństwo przypisania liter jest następujące: 37,56,70,56,37. Istnieje 5^5 możliwych słów 5 literowych. Z ciągu 256 nakładających się na siebie słów wyznaczona zostaje częstotliwość występowania każdego słowa. Następnie wyliczamy chi-kwadrat Q5 – Q4, czyli różnicy sum Pearsona gdzie Q5 to liczba słów 5 literowych a Q4 to liczba słów 4 literowych. Test przy naszej ilości 8 – bitowych liczb (655360) mogliśmy wykonać 2 razy.

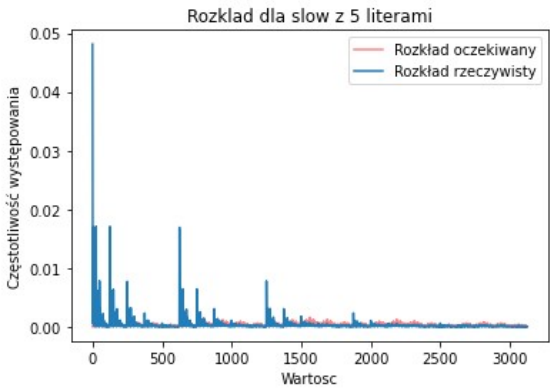
NASZ GENERATOR

Dla naszego generatora empiryczny rozkład wartości p wygląda następująco



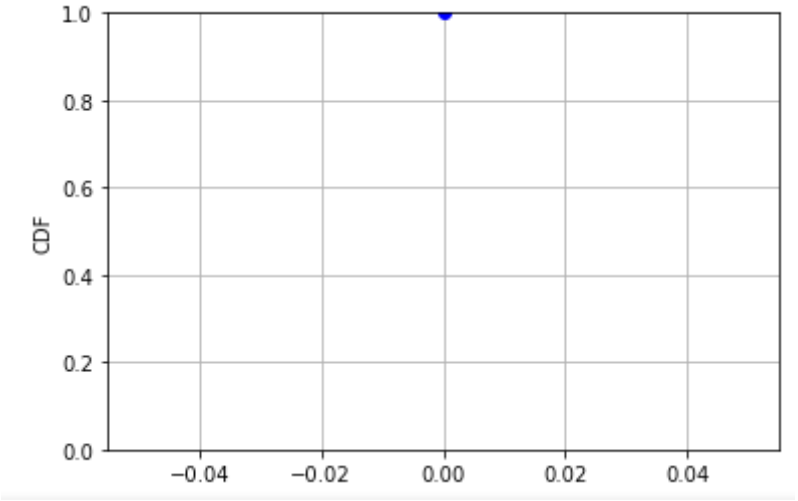
Rozkład dla 4 literowych oraz 5 literowych słów





Wartości z naszego generatora nie pokrywają się z oczekiwanym rozkładem. Widać jak duża różnica tutaj występuje.

Wykres CDF



Tak naprawdę jest to pusty wykres z jednym punktem na zerze, gdyż oba nasze przeprowadzone próby testów wynosiły 0.0.

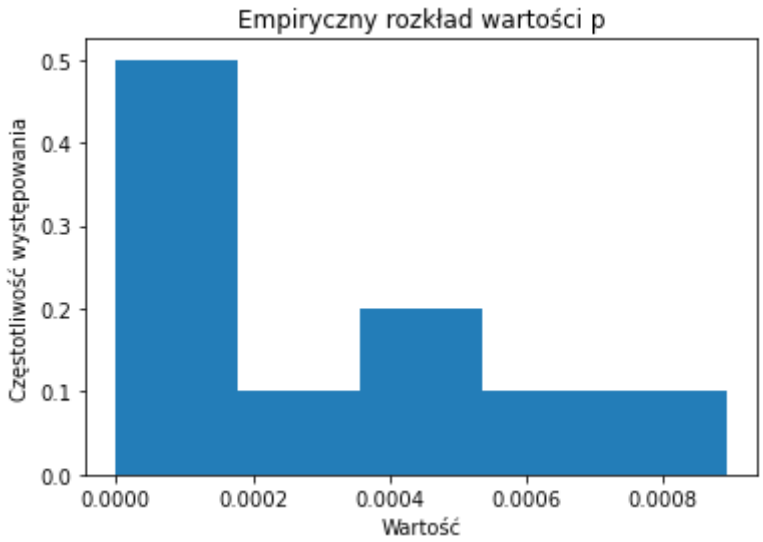
~ Uwagi

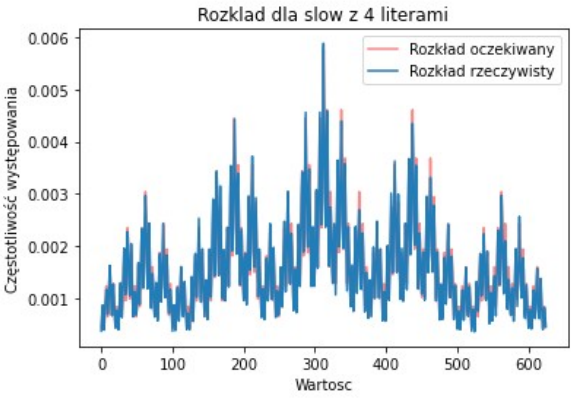
- Jak widać nasz generator potrzebuje bardzo dużej ilości zdjęć aby móc przeprowadzić test odpowiednią ilość razy. Mając 655360 wartości wykorzystując 20 zdjęć możemy przeprowadzić ten test 2 razy.

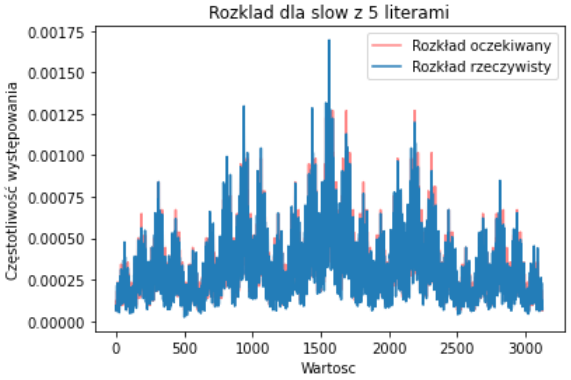
- Nasz test nie przechodzi testu Count the ones. Jest to spowodowane tym, iż nasze wartości z generatora nie są w pełni losowe.

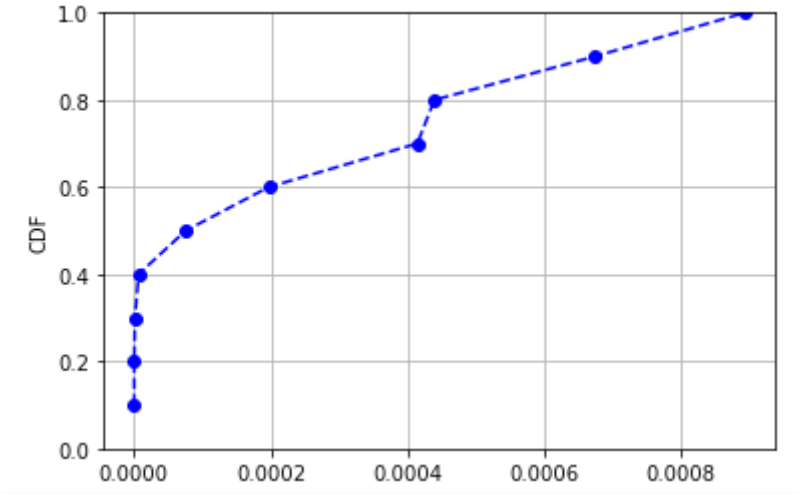
WBUDOWANY GENERATOR RANDOM

Porównując jednak nasz wynik z wynikami wbudowanego generatora liczb losowych widać bardzo duże różnice w wynikach testów. Każdy z 10 testów przeszedł bez problemów. A rozkłady występowania słów 5 oraz 4 literowych pokrywają się z oczekiwanym rozkładem.




Rozklad dla slow z 4 literami
Rozkład oczekiwany
Rozkład rzeczywisty
Częstotliwość występowania
Wartosc
0.006
0.005
0.004
0.003
0.002
0.001
0
100
200
300
400
500
600


Rozklad dla slow z 5 literami
Rozkład oczekiwany
Rozkład rzeczywisty
Częstotliwość występowania
Wartosc
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
0
500
1000
1500
2000
2500
3000


CDF
1.0
0.8
0.6
0.4
0.2
0.0
0.0000
0.0002
0.0004
0.0006
0.0008

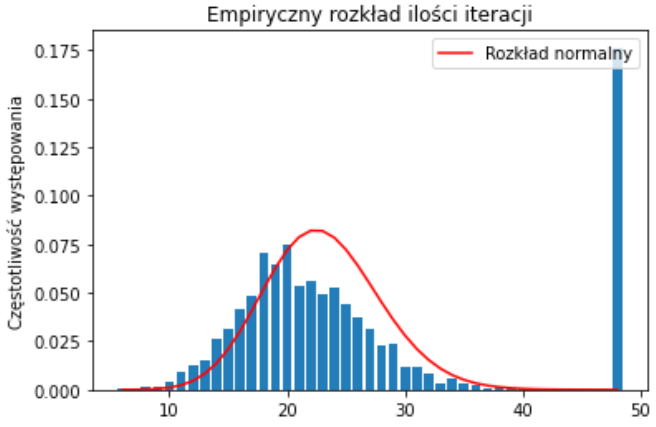
Test nr 3 – Squeeze Test

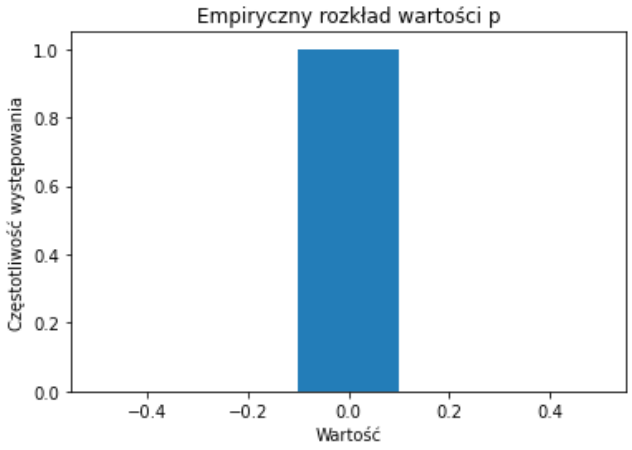
Wygenerowane liczby 32 – bitowe zostały poddane konwersji na liczby zmiennoprzecinkowe <0,1). Test zaczyna od k = 2^{31} i poszukuje liczby iteracji j niezbędnych do zmniejszenia k z wartości początkowej do 1, stosując formułę $k = \lceil kf \rceil$ gdzie f jest pobraną liczbą zmiennoprzecinkową. Korzystając z naszych wartości test przeprowadza daną liczbę powtórzeń i zlicza liczbę wystąpień poszczególnych wartości j. W przypadku gdy j < 6 wartości zostają wliczone do stanu 6, gdy j > 48 to zostają wliczone do stanu 48. Wówczas uzyskany wektor wartości poddawany jest testowi chi-kwadrat.

NASZ GENERATOR

Dla wygenerowanych 10 powtórzeń testu gdzie liczba wykonań wynosi 2000 powtórzeń wygenerowano poniższy empiryczny rozkład ilości iteracji oraz wartości p.

```
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
```





Test jest zaliczony gdy wartość p mieści się od 0,025 do 0,975. Jak widać każdy wynik testu to 0.0 - Na 10 testów 10 skończyło się niepowodzeniem. Test ogółem został niezaliczony.

~ UWAGI

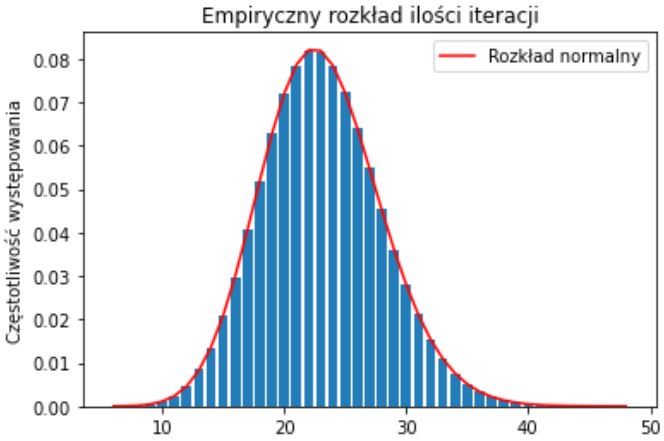
- Aby wygenerować 1000000 powtórzeń potrzeba dla naszego generatora >1000 zdjęć, zatem przyjęliśmy mniejszą liczbę powtórzeń.

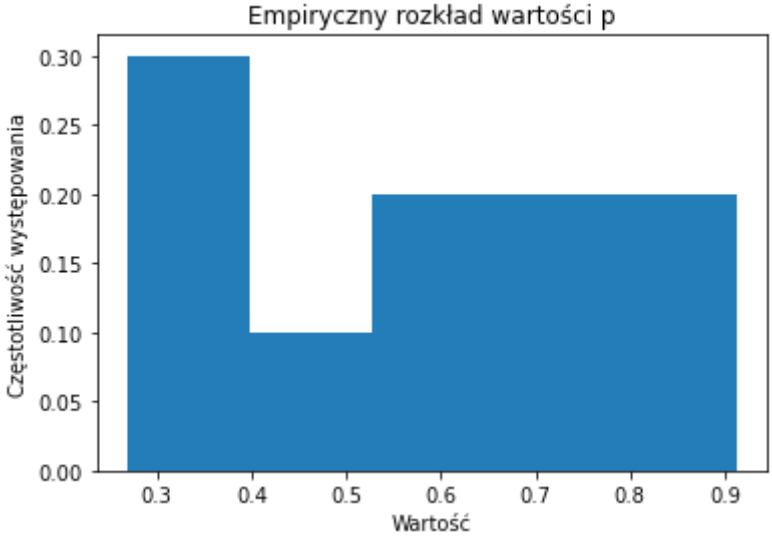
- Nie polecamy tego generatora do przeprowadzenia testów, gdzie wymagana jest duża liczba wartości.

WBUDOWANY GENERATOR RANDOM

Kod został przetestowany na liczbach randomowych, każdy z 10 testów przeszedł, zatem ogółem test został zaliczony. Poniżej jego rozkład oraz wartości p.

```
0.4201050894984744
0.725879812695262
0.8388128960525519
0.7471388785545021
0.9131548145113019
0.5933696636022608
0.2709550656456925
0.2684342382131773
0.6165368881258515
0.32262582837515924
```





PODSUMOWANIE

Generator oparty na dokumencie przez nas znalezionym nie przechodzi testów DIEHARD. Nie polecamy tego rozwiązania również z powodów długiego generowania odpowiedniej liczby wartości. Z pojedynczego zdjęcia udało się uzyskać tylko 32768 wartości (jest to kolejny bardzo duży minus tego generatora, potrzebuje strasznie dużo zasobów aby dać mu odpowiednią ilość wartości). Jest to bardzo mała liczba, w porównaniu do liczby wymaganych wartości niektórych testów np. ~4 500 000

miliona. Sama baza danych zdjęć musi być duża, a każde zdjęcie musi się różnić od pozostałych w celu jak najbardziej losowych wartości.