INSTYTUT MATEMATYKI I KRYPTOLOGII **Wydział Cybernetyki WAT**
Przedmiot: WPROWADZENIE DO KRYPTOANALIZY KLUCZA PUBLICZNEGO SPRAWOZDANIE
Temat: SITO KWADRATOWE

Wykonał: Paweł Witkowski, K5X4S1 [nr. 64024]	**Data wykonania ćwiczenia:** 01.02.2017 **Prowadzący ćwiczenie:** Kpt. Dr Mariusz Jurkiewicz

1. Wstęp teoretyczny

Dla zadanego nieparzystego N, istnieje możliwość rozbicia faktoryzacji z dużego złożonego problemu na mniejsze. W zadanym przedziale, spośród wszystkich liczb w nim się znajdujących, należy szukać B-liczb.

Przedział szukanych wartości:

$$S = \{t^2 - n : \sqrt{n} + 1 \leq t \leq 2 * (\sqrt{n} + 1)\}$$

Co skutkuje utworzeniem się funkcji pomocnicznych z zakresu:

$$F(\sqrt{n} + 1), F(\sqrt{n} + 2) \ldots . F(2 * (\sqrt{n} + 1))$$

Zadane B, dla którego szukamy liczb gładkich:

$$B = e^{\sqrt{\frac{1}{2} * logn * loglogn}}$$

Z podanego zbioru liczb pierwszych, **(2,3,5...B)** , można odrzucić liczby, które nie są resztami kwadratowymi kongruencji $t^2 \equiv N(\mathrm{mod}\, p)$.

Z takiego rozwiązania otrzymuje się dwa miejsca zerowe(wyjątek p=2), do których dodając kolejne wielokrotności **p**, uzyskuje się numery funkcji, których wartość $\mathbf{t^2-n}$ jest podzielna przez **p**. Ponadto, należy również wziąć pod uwagę kolejne potęgi każdej **p**. Podnosimy **p** do kolejnych potęg, rozwiązujemy kongruencję $t^2 \equiv N(\mathrm{mod}\, p^b)$, otrzymując znowu informacje o indeksach funkcji których wartości dzielą się znów przez **p**. W momencie, gdy tak rozwiązana kongruencja nie da rozwiązań, bądź da takie, które nie poskutkują kolejnych dzielnikiem funkcji z podanego wyżej przedziału, przechodzi się do następnej liczby Algorytm kończy swoje działanie w momencie wykorzystania wszystkich reszt kwadratowych oraz podaniem wszystkich funkcji, które udało się sfaktoryzować podanym algorytmem, wraz z bazą rozkładu danej funkcji.

2. Teoria w praktyce

Dla zadanego N należy obliczyć wymagane wartości graniczne podane w poprzednim podpunkcie. Optymalnym rozwiązaniem będzie zastosowanie wektora wektorów, gdzie "wyższy" wektor będzie zawierał wektory poszczególnych funkcji, tj ich indeks oraz bazę rozkładu. W celu uzyskania liczb pierwszych do danego **B** należy użyć odpowiedniego algorytmu, np. Sita Eratostenesa. Można odsiać te niebędące resztami kwadratowymi kongruencji $t^2 \equiv N(\mathrm{mod}\, p)$, lub zastosować odpowiedni warunek - jeśli nie uda się znaleźć pierwiasków to należy przejść do następnej liczby pierwszej. Gdy kongruencja zwróci $Alfa_p$ oraz $Beta_p$. Należy rozpocząć iteracje od $Alfa_p$ i $Beta_p$, i na wektorach odpowiadających danym funkcjom wrzucić liczbę **p**. Jeżeli uda się wykonać poprzednią operację, podnosić **p** do coraz to wyższej potęgi, rozwiązywać kongruencje oraz wrzucać na $F(Alfa_p)$, $F(Alfa_p + \mathbf{p^b})$… $F(Beta_p)$, $F(Beta_p + \mathbf{p^b})$… do momentu, gdy nic nie zostanie wrzucone na żaden z wektorów funkcji. Wtedy przejść do kolejnego **p**. Jeżeli przy rozwiązaniu kongruencji uzyska się wynik równy 0, oznacza to, że został odnaleziony trywialny dzielnik tj **p**, i można skończyć algorytm. Gdy Sito zakończy swoje działanie, wypisać tylko te funkcje, wraz z argumentem oraz wartością, które uda się sfaktoryzować elementami znajdującymi się na wektorze danej funkcji. Następnie wypisać bazę rozkładu poszczególnych wektorów.

3. Pseudokod

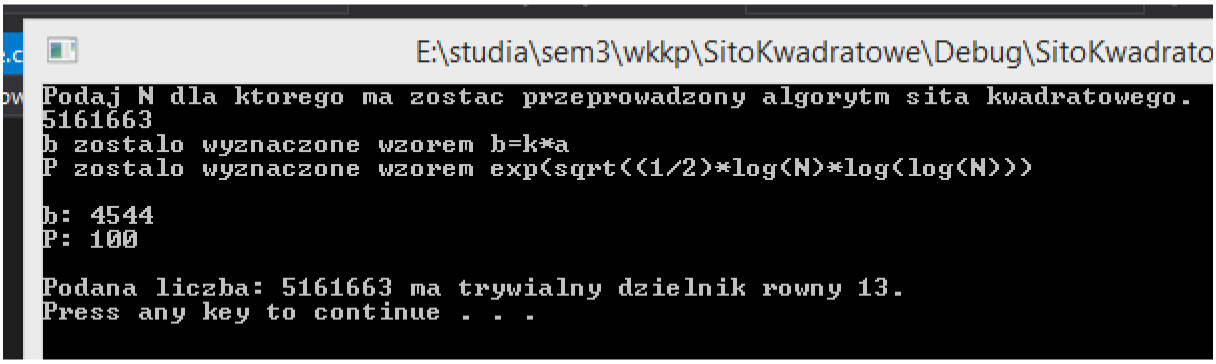
```
PrimeNumbers[] = SitoErastotenesa(PrimeBorder);
FunctionVector[][];
LowerBorder = sqrt(N) + 1;
UpperBorder = 2 * LowerBorder;
FOR ( i in PrimeNumbers[])
        IF  (N % i == 0) return i;
        IF ( i == 2 )
                DO
                        FunctionVector[ N % i + k* i]. Pushback(i);
                WHILE ( indeks <= UpperBorder);
        ELSE
                DO
                        IF (power(x,2) == N%i) FirstRoot=x;
                        IF (power(x,2) == N%i) SecondRoot=x;
                WHILE (!FirstRoot&&SecondRoot);
                DO
                        FunctionVector[ FirstRoot+ k* i]. Pushback(i);
                WHILE ( indeks <= UpperBorder);
                DO
                        FunctionVector[ SecondRoot + k* i]. Pushback(i);
                WHILE ( indeks <= UpperBorder);
        DO
                Stop=true;
                Pow = Power(i,k);
                DO
                        IF (power(x,2) == N%Pow) FirstRoot=x;
                        IF (power(x,2) == N%Pow) SecondRoot=x;
                WHILE (!FirstRoot&&SecondRoot);
                DO
                        FunctionVector[ FirstRoot+ k* i]. Pushback(i);
                        Stop =false;
                WHILE ( indeks <= UpperBorder);
                DO
```

```
                        FunctionVector[ SecondRoot + k* i]. Pushback(i);
                        Stop=false;
                WHILE ( indeks <= UpperBorder);
                K++;
                Pow=Power(i,k);
        WHILE (!stop);
FOR ( i in FuncionVector.size)
        FunctionValue = power(FuntionVector[i][0],2) - N);
        FOR ( j in FunctionVector[i].size)
                Multiplication *=FunctionVector[i][j];
        IF (FunctionValue == Multiplication) PRINT(FunctionVector[i]);
```

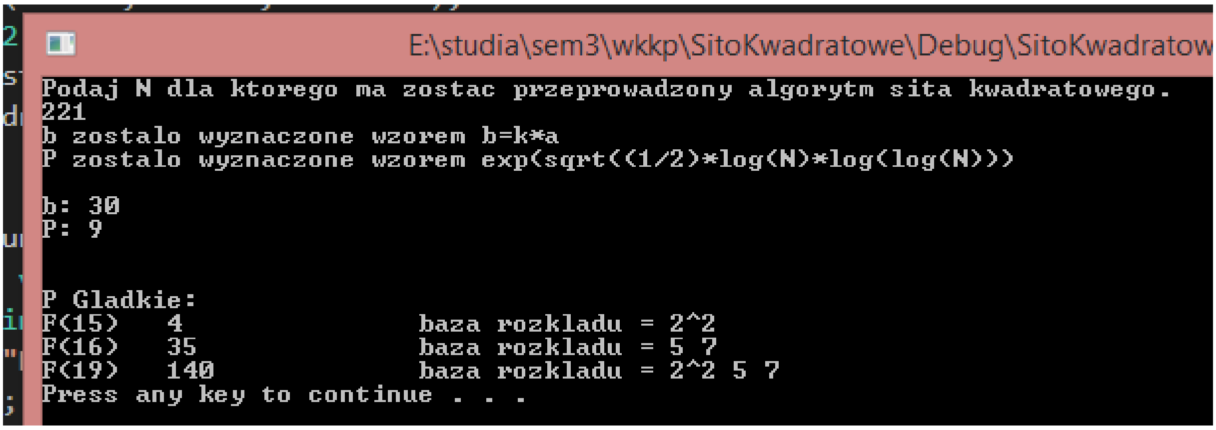
4. Wnioski

Po uruchomieniu programu, udało się zaobserwować poprawnie rozpisane Funkcje które można sfaktoryzować, oraz ich bazę rozkładu.

```
E:\studia\sem3\wkkp\SitoKwadratowe\Debug\SitoKwadratowe.exe

Podaj N dla ktorego ma zostac przeprowadzony algorytm sita kwadratowego.
381923
b zostalo wyznaczone wzorem b=2*a
P zostalo wyznaczone wzorem exp(sqrt((1/2)*log(N)*log(log(N)))

b: 1236
P: 58


P Gladkie:
F(621)  3718            baza rozkladu = 2 11 13^2
F(622)  4961            baza rozkladu = 11^2 41
F(629)  13718           baza rozkladu = 2 19^3
F(632)  17501           baza rozkladu = 11 37 43
F(657)  49726           baza rozkladu = 2 23^2 47
F(663)  57646           baza rozkladu = 2 19 37 41
F(731)  152438          baza rozkladu = 2 11 13^2 41
F(743)  170126          baza rozkladu = 2 11^2 19 37
F(798)  254881          baza rozkladu = 11 17 29 47
F(800)  258077          baza rozkladu = 17^2 19 47
F(815)  282302          baza rozkladu = 2 17 19^2 23
F(868)  371501          baza rozkladu = 13 17 41^2
F(885)  401302          baza rozkladu = 2 11 17 29 37
F(933)  488566          baza rozkladu = 2 13 19 23 43
F(959)  537758          baza rozkladu = 2 13^2 37 43
F(985)  588302          baza rozkladu = 2 11^3 13 17
F(1002) 622081          baza rozkladu = 17 23 37 43
F(1174) 996353          baza rozkladu = 17 29 43 47
F(1223) 1113806         baza rozkladu = 2 17^2 41 47
Press any key to continue . . .
```



Po odnalezieniu dzielnika trywialnego, algorytm informuje o tym.



Dla przykładu z wykładu również poprawnie wykonuje faktoryzację.

Samo sito jest wstępem do trudniejszych zagadnień, między innymi różnicy kwadratów. Dzięki niemu można w dużo szybszym czasie, dla liczb do 150 bitów, sfaktoryzować liczbę złożoną. Podane przedziały można powiększyć w celu uzyskania lepszych wyników.