Pollard rho algoritmas:

Kodas:

```
def pollard_rho(n, seed=2, f=lambda x: (x**2 + 732) % n):
    a, b = seed, seed
    factors = []
    i = 0
    isNPrime = False
    while not isNPrime:
        d = 1
        a = int(f(a) % n)
        b = int(f(f(b)) % n)
        d = gcd((a - b) % n, n)
        i += 1
        if d != 1 and d not in factors and d != n:
            factors.append(int(d))
            n = n // d
            isNPrime = isPrime(int(n))

    factors.append(int(n))
    factors.sort()
    return factors
```

Naudoju daugianarį x^2 + 732 mod(n) (n – faktorizuojamas skaičius, o daugianaris paimtas iš skaidrių) ir atlieku skaidrėse nurodytus veiksmus. Radus daliklį dalinu n iš jų tol, kol n nėra pirminis skaičius, o kai jis pasidaro pirminis pridedu jį prie daliklių. Seed – pradinės a ir b reikšmės.

Išvedimas:

```
def printPollard(n)
    p = pollard_rho(n)
    print(p)

    factorString = "{} = ".format(n)
    for factor in p:
        factorString = factorString + "{} * ".format(factor)

    factorString = factorString[:-3]
    print(factorString)
    print("-----------------------------------")

n = 11111111111111111111111111111
printPollard(n)

n = 4722366482869645141
printPollard(n)

n = 34571317791
printPollard(n)

n = 1073741823
printPollard(n)
```

Faktorizuoju jūsų duotą mažesnį skaičių (paskutinis) ir kelis savo variantus (vienas paimtas iš skaidrių), kad pademonstruoti daugiau panaudojimų.

Rezultatas:

```
[3191, 16763, 43037, 62003, 77843839397]
11111111111111111111111111111 = 3191 * 16763 * 43037 * 62003 * 77843839397
-----------------------------------
[7, 269, 587, 4272393705421]
4722366482869645141 = 7 * 269 * 587 * 4272393705421
-----------------------------------
[3, 47, 245186651]
34571317791 = 3 * 47 * 245186651
-----------------------------------
[7, 9, 151, 331, 341]
1073741823 = 7 * 9 * 151 * 331 * 341
```

Deja negalėjau faktorizuoti iš pradžių duoto skaičiaus, spėju, kad taip atsitiko, nes tas skaičius buvo dviejų didelių skaičių sandauga ir šis algoritmas arba nesugeba rasti tokių skaičių arba užtrunka daug ilgiau nei aš norėčiau laukti, taip pat jei ir randamas teisingas daliklis patikrinimas ar n yra pirminis padalinus iš šio daugiklio užtruktų labai ilgai su mano paprastu priminio skaičiaus tikrinimo algoritmu.