МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

ЛАБОРАТОРНА РОБОТА № 3.1

з дисципліни
“Інтелектуальні вбудовані системи”
на теми
“РЕАЛІЗАЦІЯ ЗАДАЧІ РОЗКЛАДАННЯ ЧИСЛА НА ПРОСТІ МНОЖНИКИ (ФАКТОРИЗАЦІЯ ЧИСЛА) ”

Виконав:
Студент групи ІП-84
Павловський В.Є.
№ ЗК: ІП-8417

Перевірив:
викладач Регіда П.Г.

Київ 2021

Основні теоретичні відомості

Факторизації лежить в основі стійкості деяких криптоалгоритмів, еліптичних кривих, алгебраїчній теорії чисел та кванових обчислень, саме тому дана задача дуже гостро досліджується, й шукаються шляхи її оптимізації.
На вхід задачі подається число n Є N, яке необхідно факторизувати. Перед виконанням алгоритму слід переконатись в тому, що число не просте. Далі алгоритм шукає перший простий дільник, після чого можна запустити алгоритм заново, для повторної факторизації.
В залежності від складності алгоритми факторизації можна розбити на дві групи:

Експоненціальні алгоритми (складність залежить експоненційно від довжини вхідного параметру);

Субекспоненціальні алгоритми.

Існування алгоритму з поліноміальною складністю – одна з найважливіших проблем в сучасній теорії чисел. Проте, факторизація з даною складністю можлива на квантовому комп'ютері за допомогою алгоритма Шора.

Метод перебору можливих дільників.

Один з найпростіших і найочевидніших алгоритмів заключається в тому, щоб послідовно ділити задане число n на натуральні числа від 1 до $\lfloor\sqrt{n}\rfloor$. Формально, достатньо ділити лише на прості числа в цьому інтервалі, але для цього необхідно знати їх множину. На практиці складається таблиця простих чисел і на вхід подаються невеликі числа (до 216), оскільки даний алгоритм має низьку швидкість роботи.

Модофікований метод факторизації Ферма.

Ідея алгоритму заключається в пошуку таких чисел A і B, щоб факторизоване число n мало вигляд: n = A2 – B2. Даний метод гарний тим, що реалізується без використання операцій ділення, а лише з операціями додавання й віднімання.

Метод факторизації Ферма.

Ідея алгоритму заключається в пошуку таких чисел A і B, щоб факторизоване число n мало вигляд: n = A2 – B2. Даний метод гарний тим, що реалізується без використання операцій ділення, а лише з операціями додавання й віднімання.

Умови завдання для варіанту

Розробити програма для факторизації заданого числа методом Ферма. Реалізувати користувацький інтерфейс з можливістю вводу даних.

Лістинг програми із заданими умовами завдання

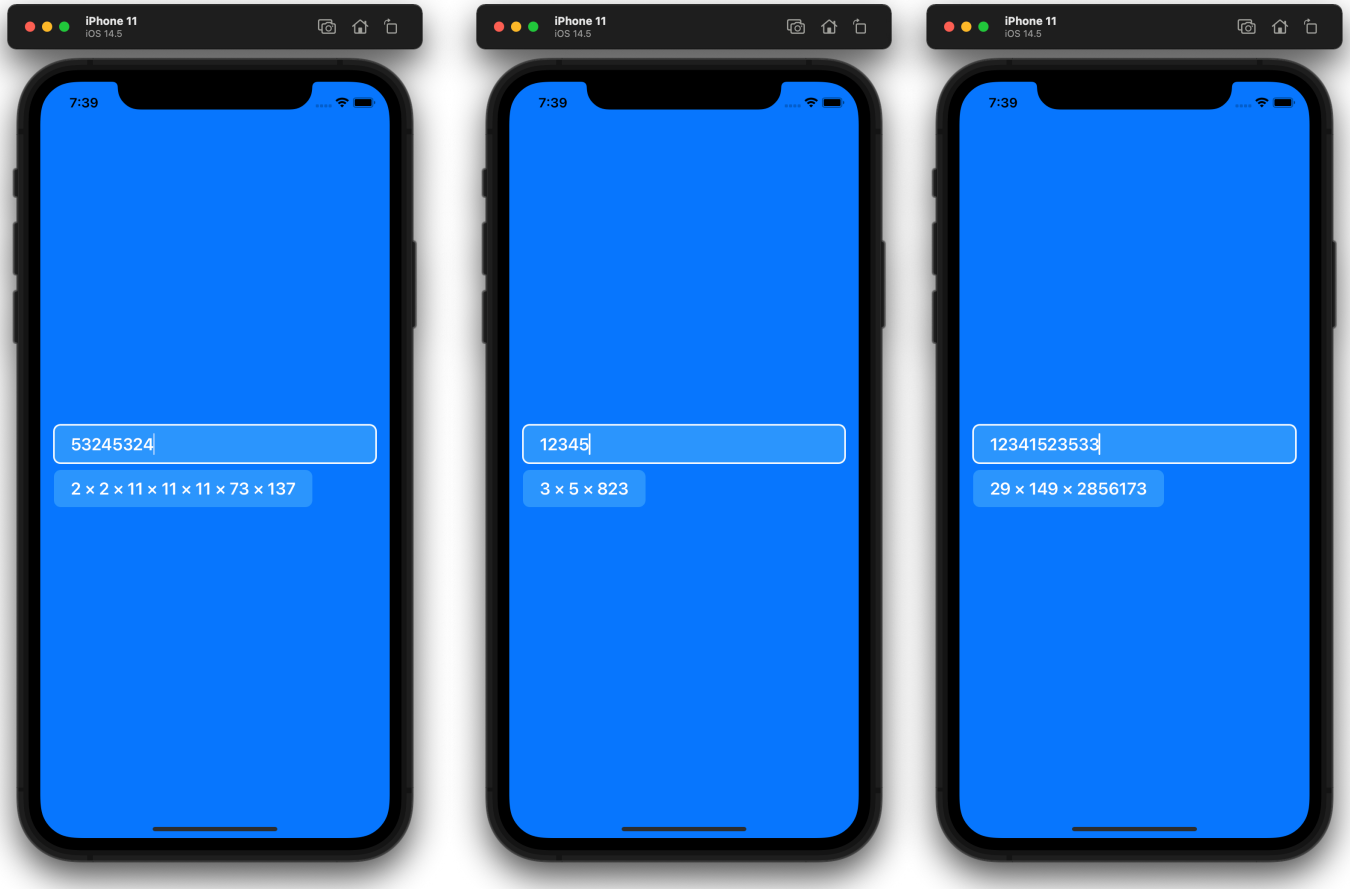
```
//
//  ContentView.swift
//  rts-lab-3
//
//  Created by Vsevolod Pavlovskyi on 16.05.2021.
//

import SwiftUI
import Combine

struct ContentView: View {

    @StateObject
    var viewModel = FermatViewModel()

    var body: some View {
        ZStack {
            Color.blue.edgesIgnoringSafeArea(.all)

            VStack(alignment: .leading) {
                TextField("Enter number", text: $viewModel.input)
                    .numberStyle(style: .bordered)
                    .font(.title2)
                    .onAppear {
                        viewModel.onAppear()
                    }

                Text(result)
                    .numberStyle()
            }
            .padding()
        }
    }

    var result: String {
        viewModel.result
            .map(String.init)
```

```
            .joined(separator: " × ")
    }

}

struct ContentView_Previews: PreviewProvider {

    static var previews: some View {
        ContentView()
    }

}

final public class FermatViewModel: ObservableObject {

    @Published var input = "1"
    @Published var result = [Int]()

    var cancellable = Set<AnyCancellable>()

    private var oldInput = ""

}

public extension FermatViewModel {

    func onAppear() {
        subscribeToNumber()
    }

}

private extension FermatViewModel {

    func compute(_ n: Int) -> [Int] {

        // If n ⩽ 3 → n is prime
        guard n > 3 else {
            return []
        }

        // If even → divide by two and continue recursively.
        guard n % 2 ≠ 0 else {
            var result = [2]

            let factorize = compute(Int(n / 2))

            guard factorize.count > 0 else {
                result.append(Int(n / 2))
```

```
                return result
            }

            result.append(contentsOf: factorize)

            return result
        }

        // Smallest number, which power 2 is more than given number.
        var x = Int(ceil(sqrt(Double(n))))

        while !perfectSquare(x * x - n) {
            x += 1
        }

        let y = Int(sqrt(Double(x * x - n)))
        let a = x - y
        let b = x + y

        var result = [Int]()

        // If number is divided not only by itself and "1", factorize recursively.
        if a ≠ 1,
           b ≠ 1 {
            let factorizedA = compute(a)
            let factorizedB = compute(b)

            // If a can be factorized → append its products.
            // If not, append only a.
            result.append(contentsOf: factorizedA.count > 0 ? factorizedA : [a])
            result.append(contentsOf: factorizedB.count > 0 ? factorizedB : [b])
        }

        return result
    }

    func perfectSquare(_ number: Int) → Bool {
        floor(sqrt(Double(number))) == sqrt(Double(number))
    }

}

private extension FermatViewModel {

    func subscribeToNumber() {
        $input
            .receive(on: RunLoop.main)
            .sink { [weak self] input in
                guard let self = self else {
```

```
                    return
                }

                let filtered = input.filter { "0123456789".contains($0) }
                if filtered ≠ input {
                    self.input = filtered
                }

                if self.oldInput == input {
                    return
                }

                self.oldInput = input

                guard let number = Int(self.input) else {
                    print("Input cannot be parsed to Int")
                    return
                }

                self.result = self.compute(number).sorted()
            }
            .store(in: &cancellable)
    }

}
```

Результати виконання програми



Висновки

Під час даної лабораторної роботи ми вивчили, що таке факторизація та розробили власний приклад алгоритму факторизації Ферма, було розроблено користувацький інтерфейс для взаємодії та наглядним використанням алгоритму.