ЛАБОРАТОРНА РОБОТА № 3
Управляючі конструкції в мові Go. Функції.

Введення та виведення інформації у консоль

Мета: ознайомитися з управляючими конструкціями мови програмування Go, навчитися користуватися функціями вводу/виводу та створювати користувацькі функції. Ознайомитись з методами генерування випадкових чисел.

Хід роботи:

20	134775813	1	2^{32}	[0, 100)	50000

Рис. 1. Вхідні дани.

Завдання 1. Розробити функцію генерування цілочислової послідовності псевдовипадкових значень (за допомогою конгруентного методу) та виконати обробку отриманого масиву даних наступним чином:

- розрахувати частоту інтервалів появи випадкових величин (інтервал дорівнює 1);
- розрахувати статистичну імовірність появи випадкових величин;
- розрахувати математичне сподівання випадкових величин;
- розрахувати дисперсію випадкових величин;
- розрахувати середньоквадратичне відхилення випадкових величин.

Завдання 2. Розробити функцію генерування послідовності псевдовипадкових дійсних значень (за допомогою конгруентного методу).

Листинг програми:

Листинг програми:

```
package main

import (
    "fmt"
    "math"
)

const (
    a = uint64(134775813)
    c = uint64(1)
    m = uint64(1) << 32
    K = 50000
)
```

					ДУ «Житомирська політехніка».25.121.20.000 – Лр3			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Риженко Я.В.			Звіт з лабораторної роботи	Літ.	Арк.	Аркушів
Перевір.		Петросян Р.В.					1	3
Керівник						ФІКТ Гр. ІПЗ-23-1		
Н. контр.								
Зав. каф.								

```go
func GenerateIntSequence(seed uint64) []uint64 {
    x := seed
    seq := make([]uint64, K)

    for i := 0; i < K; i++ {
       x = (a*x + c) % m
       seq[i] = x % 100
    }
    return seq
}

func GenerateFloatSequence(seed uint64) []float64 {
    x := seed
    seq := make([]float64, K)

    for i := 0; i < K; i++ {
       x = (a*x + c) % m
       seq[i] = (float64(x) / float64(m)) * 100
    }
    return seq
}

func Analyze(seq []uint64) {
    freq := make([]int, 100)

    for _, v := range seq {
       freq[v]++
    }
    prob := make([]float64, 100)
    for i := 0; i < 100; i++ {
       prob[i] = float64(freq[i]) / float64(len(seq))
    }

    var mean float64
    for i := 0; i < 100; i++ {
       mean += float64(i) * prob[i]
    }

    var dispersion float64
    for i := 0; i < 100; i++ {
       dispersion += math.Pow(float64(i)-mean, 2) * prob[i]
    }

    std := math.Sqrt(dispersion)

    fmt.Println("=== Результати аналізу ===")
    fmt.Printf("Математичне сподівання: %.6f\n", mean)
    fmt.Printf("Дисперсія: %.6f\n", dispersion)
    fmt.Printf("СКВ: %.6f\n\n", std)

    fmt.Println("=== Ймовірності появи значень ===")
    for i := 0; i < 100; i++ {
       fmt.Printf("%2d : P = %.6f\n", i, prob[i])
    }
}
func main() {
    seq := GenerateIntSequence(1)
    Analyze(seq)

    floatSeq := GenerateFloatSequence(1)
    fmt.Println("Перші 5 дійсних значень [0;100):", floatSeq[:5])
}
```

Результат виконання:

```
PS D:\Work\GO\lab3\ex> go run .
=== Результати аналізу ===
Математичне сподівання: 49.418960
Дисперсія: 837.775353
СКВ: 28.944349

=== Ймовірності появи значень ===
 0 : P = 0.010320
 1 : P = 0.010400
 2 : P = 0.010480
 3 : P = 0.010280
 4 : P = 0.010260
 5 : P = 0.010880
 6 : P = 0.009820
 7 : P = 0.010020
 8 : P = 0.009640
 9 : P = 0.010160
10 : P = 0.009660
11 : P = 0.010160
12 : P = 0.009440
13 : P = 0.009440
14 : P = 0.010600
15 : P = 0.010040
16 : P = 0.010340
17 : P = 0.010220
18 : P = 0.010380
```

Рис. 1. Завдання 1.

```
Перші 5 дійсних значень [0;100): [3.1379939522594213 86.10484672244638 20.258096512407064 27.29212671983987 67.16544185765088]
```

Рис. 2. Завдання 2.

Посилання на репозиторій - https://github.com/JanRizhenko/GoLang-sigma-practice

Висновок: У ході роботи було розглянуто основні управляючі конструкції мови Go, що дозволяють будувати логіку програми та керувати її виконанням. Було опановано роботу з консольним введенням і виведенням, що є фундаментом для взаємодії користувача з програмою. Окрему увагу приділено створенню власних функцій, які допомагають структурувати код, робити його зрозумілішим і придатним до повторного використання. Також було ознайомлено з принципами генерування випадкових чисел, зокрема зі способами ініціалізації та використання генераторів. У результаті сформовано цілісне розуміння того, як поєднуються логічні конструкції, функції та взаємодія з користувачем у практичних задачах програмування на Go.