**Лабораторна робота №1**

Нормативно-правова база захисту

інформації. Обробка тексту

Мета: закріпити знання про основні аспекти та нормативно-правову базу захисту інформації, про сервісні служби та механізми захисту, про порушення захисту, про класи таємності; набути навичок статистичної обробки тексту.

**Завдання**

1. Написати реферат на тему «Закони України «Про інформацію» та «Про захист інформації в автоматизованих системах»». Особливу увагу приділити статті N Закону України «Про захист інформації в автоматизованих системах», де N – номер варіанту. Об’єм реферату не менше 5000 знаків.
2. Написати програму підрахунку частоти українських букв та пар букв у написаному рефераті.
3. За допомогою розробленої програми побудувати гістограму повторень перших 2000 знаків у тексті за зобразити їх у кольоровому вигляді.

**Реферат на тему Закони України «Про інформацію» та «Про захист інформації в автоматизованих системах»**

Зміст

1. Вступ
2. Закон України «Про інформацію»
3. Закон України «Про захист інформації в автоматизованих система»
4. Висновки
5. Вступ

У наш час ми все не можемо уявити своє життя без комп’ютерної техніки та інтернету. Мабуть вже немає людей, у яких немає телефону, але практично всі мають як мінімум комп’ютер та телефон.

Крім того сучасний телефон – це все давно не просто засіб для зв’язку. NFC, FaceId, Fingerprint тощо, навіть телефон стає гаманцем та сховищем великої кількості приватної інформації.

Якщо подивитись на Інтернет – це невід’ємна частина нашого життя. Ми купуємо в інтернеті, ми навіть отримуємо доступ до банківського рахунку через інтернет.

Соціальні мережі також невід’ємна частина нашого життя, і в той же час в кожній соціальній мережі зберігається незліченна кількість інформації, до якої повинні мати доступ лише певні люди.

Тому очевидно, що захист інформації – це дуже важливо. Якщо хтось зможе обійти сучасні засоби захисту інформації, то в його можливостях буде нанести стільки же шкоди, скільки, мабуть, від атомної бомби, тобто, це може ввести світ у хаос.

Крім того, захист інформації має гарантуватись законом. Інакше багато хто буде відноситись до цього несерйозно, або використовувати приватну інформацію (наприклад, своїх користувачів) у своїх цілях.

В Україні також є закони, що описують захист інформації. Наприклад: Закон України «Про інформацію» та «Про захист інформації в автоматизованих система»

1. Закон України «Про інформацію»

Закон України «Про інформацію» дає визначення що таке інформація, та захист інформації, які є види інформації, тощо.

Ось основні терміни, що показані у документі (ст1):

Документ - матеріальний носій, що містить інформацію, основними функціями якого є її збереження та передавання у часі та просторі;

Захист інформації - сукупність правових, адміністративних, організаційних, технічних та інших заходів, що забезпечують збереження, цілісність інформації та належний порядок доступу до неї;

Інформація - будь-які відомості та/або дані, які можуть бути збережені на матеріальних носіях або відображені в електронному вигляді;

Суб'єкт владних повноважень - орган державної влади, орган місцевого самоврядування, інший суб'єкт, що здійснює владні управлінські функції відповідно до законодавства, у тому числі на виконання делегованих повноважень.

Також у законі показано основні принципи інформаційних відносин:

1. гарантованість права на інформацію;
2. відкритість, доступність інформації, свобода обміну інформацією;
3. достовірність і повнота інформації;
4. свобода вираження поглядів і переконань;
5. правомірність одержання, використання, поширення, зберігання та захисту інформації;
6. захищеність особи від втручання в її особисте та сімейне життя.

Важливою, є також ст. 5, що каже:

1. Кожен має право на інформацію, що передбачає можливість вільного одержання, використання, поширення, зберігання та захисту інформації, необхідної для реалізації своїх прав, свобод і законних інтересів.

Реалізація права на інформацію не повинна порушувати громадські, політичні, економічні, соціальні, духовні, екологічні та інші права, свободи і законні інтереси інших громадян, права та інтереси юридичних осіб.

Також, згідно з цим законом є такі вид інформації (ст 10.):

За змістом інформація поділяється на такі види:

1. інформація про фізичну особу;
2. інформація довідково-енциклопедичного характеру;
3. інформація про стан довкілля (екологічна інформація);
4. інформація про товар (роботу, послугу);
5. науково-технічна інформація;
6. податкова інформація;
7. правова інформація;
8. статистична інформація;
9. соціологічна інформація;
10. інші види інформації.

Інформація про особу, мабуть, одна з найважливіших з погляду захисту інформації. Ось що каже стаття 11 другого розділу:

Не допускаються збирання, зберігання, використання та поширення конфіденційної інформації про особу без її згоди, крім випадків, визначених законом, і лише в інтересах національної безпеки, економічного добробуту та захисту прав людини.

Кожному забезпечується вільний доступ до інформації, яка стосується його особисто, крім випадків, передбачених законом.

1. Закон України «Про захист інформації в інформаційно-телекомунікаційних системах»

В першу чергу у законі приводяться визначення основних термінів. Ось визначення найважливіших з них:

обробка інформації - вся сукупність операцій (збирання, введення, записування, перетворення, зчитування, зберігання, знищення, реєстрація), що здійснюються за допомогою технічних і програмних засобів, включаючи обмін по каналах передачі даних;

захист інформації - сукупність організаційно-технічних заходів і правових норм для запобігання заподіянню шкоди інтересам власника інформації чи АС та осіб, які користуються інформацією; несанкціонований доступ - доступ до інформації, що здійснюється з порушенням встановлених в АС правил розмежування доступу;

витік інформації - результат дій порушника, внаслідок яких інформація стає відомою (доступною) суб'єктам, що не мають права доступу до неї;

втрата інформації - дія, внаслідок якої інформація в АС перестає існувати для фізичних або юридичних осіб, які мають право власності на неї в повному чи обмеженому обсязі;

підробка інформації - навмисні дії, що призводять до перекручення інформації, яка повинна оброблятися або зберігатися в АС;

блокування інформації - дії, наслідком яких є припинення доступу до інформації;

порушення роботи АС - дії або обставини, які призводять до спотворення процесу обробки інформації.

Цікавою є також стаття 5 цього закону (Гарантія юридичного захисту):

Суб'єкти права власності, визначені авторським правом або договірними відносинами, мають право на юридичний захист від заподіяння шкоди власнику інформації чи АС внаслідок навмисної чи ненавмисної втрати, знищення, підроблення, спотворення, блокування інформації та інших неправомірних дій.

Ця стаття показує, що захист інформації – це не просто голосна назва, вона показує, що злочинні дії пов’язанні з інформацією це такі самі злочини, як і у реальному житі, і за них може бути реальна відповідальність.

Висновок:

Зрозуміло, що законодавча база України з приводу захисту інформації не ідеальна. Крім того, на практиці часто деякі моменти, що вказані у законі не виконуються. Але все одно закон Україні «Про інформацію» та закон Україні «Про захист інформації в інформаційно-телекомунікаційних системах» показують, що захист інформації невід’ємна частина будь-якої автоматизованої системи.

1. Текст програми

export interface ICharFrequency {

   map: Map<string, number>;

   len: number;

}

export interface ICharFrequencyOptions {

   ignoreCase?: boolean;

   spaces?: boolean;

   digits?: boolean;

   punctuation?: boolean;

}

const DEF\_FREQUENCY\_OPTIONS: ICharFrequencyOptions = {

   ignoreCase: true,

   spaces: false,

   digits: false,

   punctuation: false,

}

/\*\*

 \* Calculate character frequency in the text

 \* @param text

 \* @param options

 \* @returns map of characters and character frequency (in range [0, 1])

 \*/

export default function calcCharFrequency(text: string, options?: ICharFrequencyOptions): ICharFrequency {

   options = { ...DEF\_FREQUENCY\_OPTIONS, ...options || {} };

   text = prepareText(text, options);

   return {

      map: \_calcFrequency(text.split('')),

      len: text.length,

   }

}

function prepareText(text: string, options: ICharFrequencyOptions): string {

   const { ignoreCase, spaces, digits, punctuation } = options;

   text = text.trim();

   if (ignoreCase) text = text.toLowerCase();

   if (!spaces) text = text.replace(/\s+/g, '')

   if (!digits) text = text.replace(/\d+/g, '');

   if (!punctuation) {

      text = text.replace(/[.,\/#!$\?%\^&\\*";:{}=\-\_`~()\[\]]+/g, '');

   }

   return text;

}

export interface ICharPairsFrequency {

   map: Map<string, number>;

   len: number;

}

/\*\*

 \* Calculate frequency of two-character sequences in the text

 \* @param text

 \* @returns map of two-character sequences and its frequency (in range [0, 1])

 \*/

export default function calcCharPairFrequency(text: string): ICharPairsFrequency {

   text = text.toLowerCase().replace(/[^a-zа-яёїієґ]+/g, '');

   const pairs = toPairs(text);

   return {

      map: \_calcFrequency(pairs),

      len: pairs.length,

   }

}

function toPairs(text: string): string[] {

   const chars = text.split('');

   const pairs: string[] = [];

   const len = chars.length;

   for (let i = 0; i < len - 1; i++) {

      pairs.push(chars[i] + chars[i + 1]);

   }

   return pairs;

}

export default function \_calcFrequency(arr: any[]): Map<string, number> {

   const charsMap: Map<string, number> = new Map();

   const frequencyMap: Map<string, number> = new Map();

   arr.forEach((char) => {

      if (charsMap.has(char)) {

         charsMap.set(char, charsMap.get(char) + 1);

      } else {

         charsMap.set(char, 1);

      }

   });

   charsMap.forEach((value, key) => {

      frequencyMap.set(key, value / arr.length);

   });

   return frequencyMap;

}

import { debounce } from 'throttle-debounce';

import 'normalize.scss/normalize.scss';

import './index.scss';

import Editor from './scripts/Editor';

import Frequency from './scripts/Frequency';

import Res from './scripts/Res';

const $: { [type: string]: HTMLElement } = {};

$.root = document.querySelector('.app');

$.editor = $.root.querySelector('.app-editor')

$.res = $.root.querySelector('.app-res');

const toolbarForm: HTMLFormElement = $.root.querySelector('.app-toolbar\_\_form');

const frequency = new Frequency();

const editor = new Editor($.editor);

const res = new Res($.res, frequency);

const update = debounce(500, () => {

    updateFrequency();

    editor.setHighlight(toolbarForm.highlight.checked);

    editor.setHighlightMap(frequency.charColors);

});

update();

initEvents();

function initEvents() {

    editor.events.addListener('input', () => {

        update();

    });

    toolbarForm.highlight.addEventListener('change', () => {

        editor.setHighlight(toolbarForm.highlight.checked);

    });

    toolbarForm.case.addEventListener('change', update);

    toolbarForm.spaces.addEventListener('change', update);

    toolbarForm.digits.addEventListener('change', update);

    toolbarForm.punctuation.addEventListener('change', update);

}

function updateFrequency() {

    frequency.update(editor.getText(), {

        ignoreCase: toolbarForm.case.checked,

        spaces: !toolbarForm.spaces.checked,

        digits: !toolbarForm.digits.checked,

        punctuation: !toolbarForm.punctuation.checked,

    });

}

import calcCharPairsFrequency, { ICharPairsFrequency } from './algorithms/char-frequency/calc-char-pair-frequency';

import calcCharFrequency, { ICharFrequency, ICharFrequencyOptions } from './algorithms/char-frequency/calc-char-frequency';

import Events from './Events';

export default class Frequency {

   public events = new Events();

   public char: ICharFrequency;

   public pair: ICharPairsFrequency;

   public charColors: Map<string, string> = new Map();

   public pairColors: Map<string, string> = new Map();

   public mostFrequentChar = { key: '', val: 0 };

   public mostFrequentPair = { key: '', val: 0 };

   public update(text: string, options: ICharFrequencyOptions) {

      this.char = calcCharFrequency(text, options);

      this.pair = calcCharPairsFrequency(text);

      this.updateMostFrequentChar();

      this.updateMostFrequentPair();

      this.updateCharColors();

      this.updatePairColors();

      this.events.emit('update');

   };

   private updateCharColors() {

      this.charColors.clear();

      this.char.map.forEach((val, key) => {

         const v = val \* 255 / this.mostFrequentChar.val;

         this.charColors.set(key, `rgb(${v},0,0)`);

      });

   }

   private updatePairColors() {

      this.pairColors.clear();

      this.pair.map.forEach((val, key) => {

         const v = val \* 255 / this.mostFrequentPair.val;

         this.pairColors.set(key, `rgb(${v},0,0)`);

      });

   }

   private updateMostFrequentChar() {

      this.mostFrequentChar = { key: '', val: 0 };

      this.char.map.forEach((val, key) => {

         if (this.mostFrequentChar.val >= val) return;

         this.mostFrequentChar.val = val;

         this.mostFrequentChar.key = key;

      });

   }

   private updateMostFrequentPair() {

      this.mostFrequentPair = { key: '', val: 0 };

      this.pair.map.forEach((val, key) => {

         if (this.mostFrequentPair.val >= val) return;

         this.mostFrequentPair.val = val;

         this.mostFrequentPair.key = key;

      });

   }

}

import { throttle } from 'throttle-debounce';

import Events from './Events';

interface IEditorElements {

   [name: string]: HTMLElement,

   realText?: HTMLTextAreaElement,

}

export default class Editor {

   private MAX\_HIGHLIGHT\_LEN = 5000;

   private MAX\_LIVE\_HIGHLIGHT\_LEN = 1000;

   private \_value = '';

   private $: IEditorElements = {};

   private \_decTextUpdatingFrame = 0;

   private \_highlight = false;

   private \_highlightMap: Map<string, string> = new Map();

   public events = new Events();

   constructor(private root: HTMLElement) {

      this.init();

   }

   private init() {

      this.getElements();

      this.initValue();

      this.setHighlight(this.\_highlight);

      this.initEvents();

   }

   public setHighlight(val: boolean) {

      this.\_highlight = val;

      this.updateDecText();

      const type = (val) ? 'remove' : 'add';

      this.root.classList[type]('app-editor--no-highlight');

   }

   public setHighlightMap(highlightMap: Map<string, string>) {

      this.\_highlightMap = highlightMap;

      this.updateDecText();

   }

   public getText() {

      return this.\_value;

   }

   private initEvents() {

      this.$.realText.addEventListener('input', () => {

         if (this.\_value.length > this.MAX\_LIVE\_HIGHLIGHT\_LEN) {

            return;

         }

         this.startDecTextUpdating();

         setTimeout(() => this.stopDecTextUpdating(), 1000);

      });

      this.$.realText.addEventListener('input', throttle(300, () => {

         if (this.\_value.length > this.MAX\_LIVE\_HIGHLIGHT\_LEN) {

            this.updateDecTextScrolls();

            this.readRealText();

         }

      }));

      this.$.realText.addEventListener('keyup', throttle(300, () => {

         this.stopDecTextUpdating();

         this.updateDecTextScrolls();

         this.readRealText();

         this.events.emit('input');

      }));

      this.$.realText.addEventListener('scroll', () => {

         this.updateDecTextScrolls();

      });

   }

   /\*\* Start automatic dec text updating \*/

   private startDecTextUpdating() {

      if (this.\_decTextUpdatingFrame) return;

      const editor = this;

      this.\_decTextUpdatingFrame = requestAnimationFrame(function tik() {

         editor.readRealText();

         editor.\_decTextUpdatingFrame = requestAnimationFrame(tik);

      });

   }

   private stopDecTextUpdating() {

      cancelAnimationFrame(this.\_decTextUpdatingFrame);

      this.\_decTextUpdatingFrame = 0;

   }

   private readRealText() {

      this.\_value = this.$.realText.value;

      this.updateDecText();

   }

   private updateDecTextScrolls() {

      this.$.decText.scrollTop = this.$.realText.scrollTop;

      this.$.decText.scrollLeft = this.$.realText.scrollLeft;

   }

   private updateDecText() {

      if (!this.\_highlight || this.\_value.length > this.MAX\_HIGHLIGHT\_LEN) {

         this.$.decText.innerHTML = this.\_value;

         return;

      }

      this.$.decText.innerHTML = this.\_value.split('').map((item, i) => {

         const map = this.\_highlightMap;

         item = this.escape(item);

         if (!map.has(item) && !map.has(item.toLowerCase())) {

            return item;

         }

         const color = map.get(item) || map.get(item.toLowerCase());

         return `<span style="color: ${color}">${item}</span>`;

      }).join('');

   }

   private escape(str) {

      return str

         .replace(/&/g, '&amp;')

         .replace(/</g, '&lt;')

         .replace(/>/g, '&gt;')

         .replace(/"/g, '&quot;')

         .replace(/'/g, '&#039;')

   }

   private initValue() {

      this.\_value = this.$.realText.value;

   }

   private getElements() {

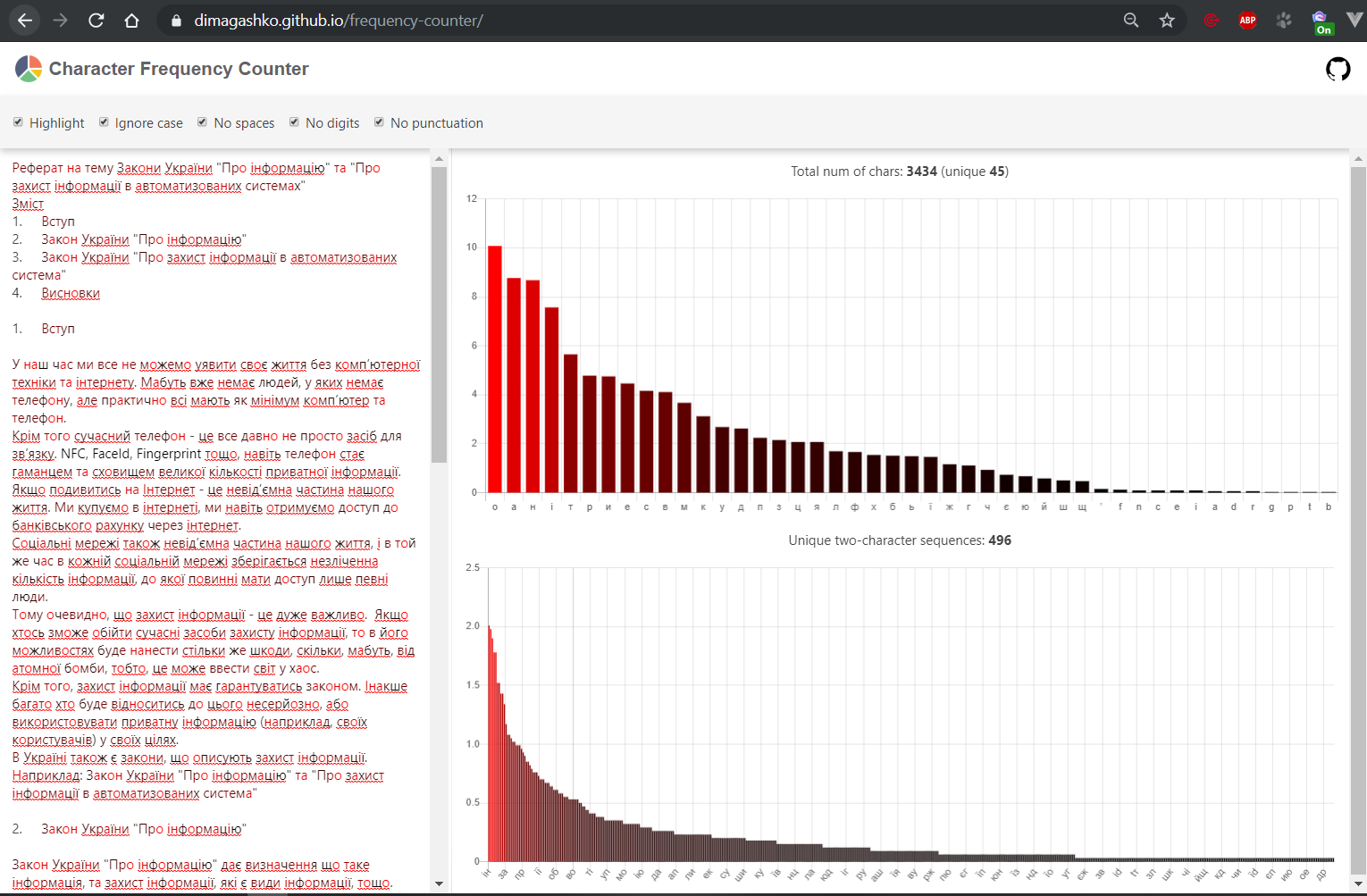
      this.$.decText = this.root.querySelector('.app-editor\_\_dec-text');

      this.$.realText = this.root.querySelector('.app-editor\_\_real-text');

   }

}

1. Результат виконання програми



**Висновок:** на цій лабораторній роботі я ознайомився з предикатами введення-виведення у мові Prolog; вивчив та одержав практичні навички з формування та змінною бази даних програми.