Лабораторна робота № 1

Тема: Реалізація рекурсивних алгоритмів опрацювання структур даних.

Мета роботи: Набуття практичних навичок роботи з рекурсивними функціями.

## Хід роботи

Приклад 1.1. Розглянемо рекурсивну підпрограму для обчислення чисел Фібоначчі.

Лістинг 1.1

```
//Чавурська Єлизавета, ЛР1, Варіант 20
#include <iostream>
using namespace std;
// Рекурсивна функція
void Rec(int a) {
	if (a > 0) {
		Rec(a - 1);
	}
	cout << a << endl;
	setlocale(LC_CTYPE, "UKR");
}
int main() {
	Rec(3);  // приклад виклику
	// Виведення підпису
	cout << "Чавурська Єлизавета, ЛР1, Варіант 20" << endl;
	return 0;
}
```

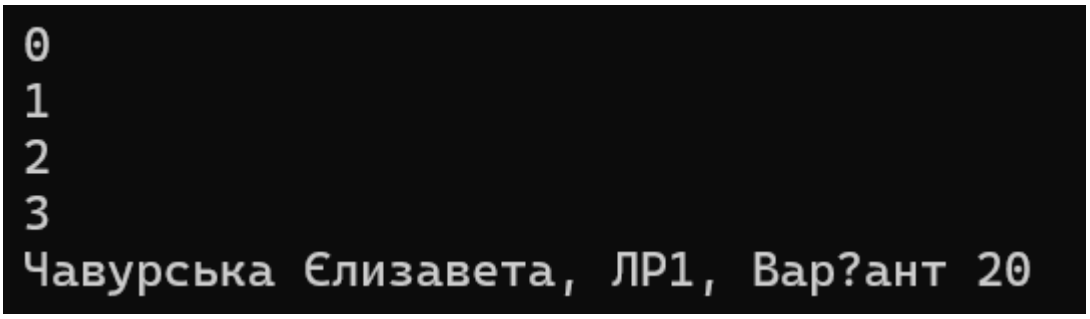



Рисунок 1.1 – Вікно результатів програми

Приклад 1.2. Описати рекурсивну функцію для визначення факторіалу числа. Факторіал визначається рекурсивною функцією:

$$F(0) = 1,$$

$$F(n) = n \cdot F((n-1), n \geq 1.$$

Лістинг 1.2 – Виконання прикладу 2 на мові С++

```
//Чавурська Єлизавета, ЛР1, Варіант 20
#include <iostream>
using namespace std;
int Fact(int n) {
	setlocale(LC_CTYPE, "Ukr");
	if (n == 0) // Термінальна гілка
		return 1; // Тривіальне значення
	else
		return n * Fact(n - 1); // Рекурсивний виклик
}
int main() {
	int n;
	cout << "n = ";
	cin >> n;
	cout << n << "! = " << Fact(n) << endl;
	cout << "Чавурська Єлизавета, ЛР1, Варіант 20, Приклад 2" << endl;
	return 0;
}
```

```
n = 3
3! = 6
Чавурська Єлизавета, ЛР1, Вар?ант 20, Приклад 2
```

Рисунок 1.2 – Вікно результатів Прикладу 2

Приклад 1.3. Маємо функцію, що розкладається у ряд:

$$s = \prod_{x=1}^{n} \frac{x}{e^x - x^2}.$$

Лістинг 1.2. Обчислення суми ряду

```
//Чавурська Єлизавета, ЛР1, Варіант 20
#include <iostream>
#include <cmath>
using namespace std;
// Прототипи функцій
double func1(int n);   // Нерекурсивна функція
double func2(int n);   // Рекурсивна функція
int main() {
	setlocale(LC_CTYPE, "Ukr");
	int n;  // n – кількість елементів ряду
	cout << "n = ";
```

```
    cin >> n;  // Ввід n
    cout << "Нерекурсивна функція = " << func1(n) << endl;
    cout << "Рекурсивна функція = " << func2(n) << endl;
    cout << "Чавурська Єлизавета, ЛР1, Варіант 20, Приклад 3" << endl;
    return 0;
}
// Нерекурсивна функція
double func1(int n) {
    double s = 1.0;
    for (int x = 1; x <= n; x++) {
        s *= x / (exp(x) - x * x);
    }
    return s;
}
// Рекурсивна функція
double func2(int n) {
    if (n < 1)
        return 1.0; // умова виходу з рекурсії
    else
        return n / (exp(n) - n * n) * func2(n - 1); // рекурсивний виклик
}
```

```
n = 2
Нерекурсивна функц?я = 0.343445
Рекурсивна функц?я = 0.343445
Чавурська Єлизавета, ЛР1, Вар?ант 20, Приклад 3
```

Рисунок 1.3. – Вікно результатів програми

Завдання для виконання:

Проаналізувати описані приклади 1.1 та 1.2; виконати програми, що викликають рекурсивні функції із не менш як трьома різними вхідними даними.

Результати занести у звіт.

1. За прикладом 1.3 розробити алгоритм обчислення величини, описаної у Вашому варіанті без використання та із використанням рекурсивної функції.

2. Написати функції обчислення суми чи добутку без використання рекурсії та з її використанням на будь якій мові програмування, крім наведеної у прикладі.

| | |
|---|---|
| **20** | $z = \sum_{x=1}^{n} (\sin x + x)$ |

Лістинг 1.4 – Код програми

```
//Чавурська Єлизавета, ЛР1, Варіант 20
#include <iostream>
#include <cmath>
#include <iomanip>
using namespace std;
// Прототипи функцій
double func1(int n);   // Нерекурсивна функція
double func2(int n);   // Рекурсивна функція
int main() {
	setlocale(LC_CTYPE, "Ukr");
	int n;
	cout << "n = ";
	cin >> n;
	cout << fixed << setprecision(2);
	cout << "Нерекурсивна функція = " << func1(n) << endl;
	cout << "Рекурсивна функція = " << func2(n) << endl;
	cout << "Чавурська Єлизавета, ЛР1, Варіант 20, Приклад 2" << endl;
	return 0;
}

// Нерекурсивна функція: z = Σ(sin(x) + x)
double func1(int n) {
	double z = 0.0;
	for (int x = 1; x <= n; x++) {
		z += sin(x) + x;
	}
	return z;
}
// Рекурсивна функція: z = Σ(sin(x) + x)
double func2(int n) {
	if (n < 1)
		return 0.0; // Умова виходу
	else
		return sin(n) + n + func2(n - 1); // Рекурсивний виклик
}
```

3. Виконати програму із не менш як трьома різними вхідними даними.

Результати занести у звіт.

```
n = 2
Нерекурсивна функц?я = 4.75
Рекурсивна функц?я = 4.75
Чавурська Єлизавета, ЛР1, Вар?ант 20
```

Рисунок 1.4 – Вікно результатів програми 1 приклад

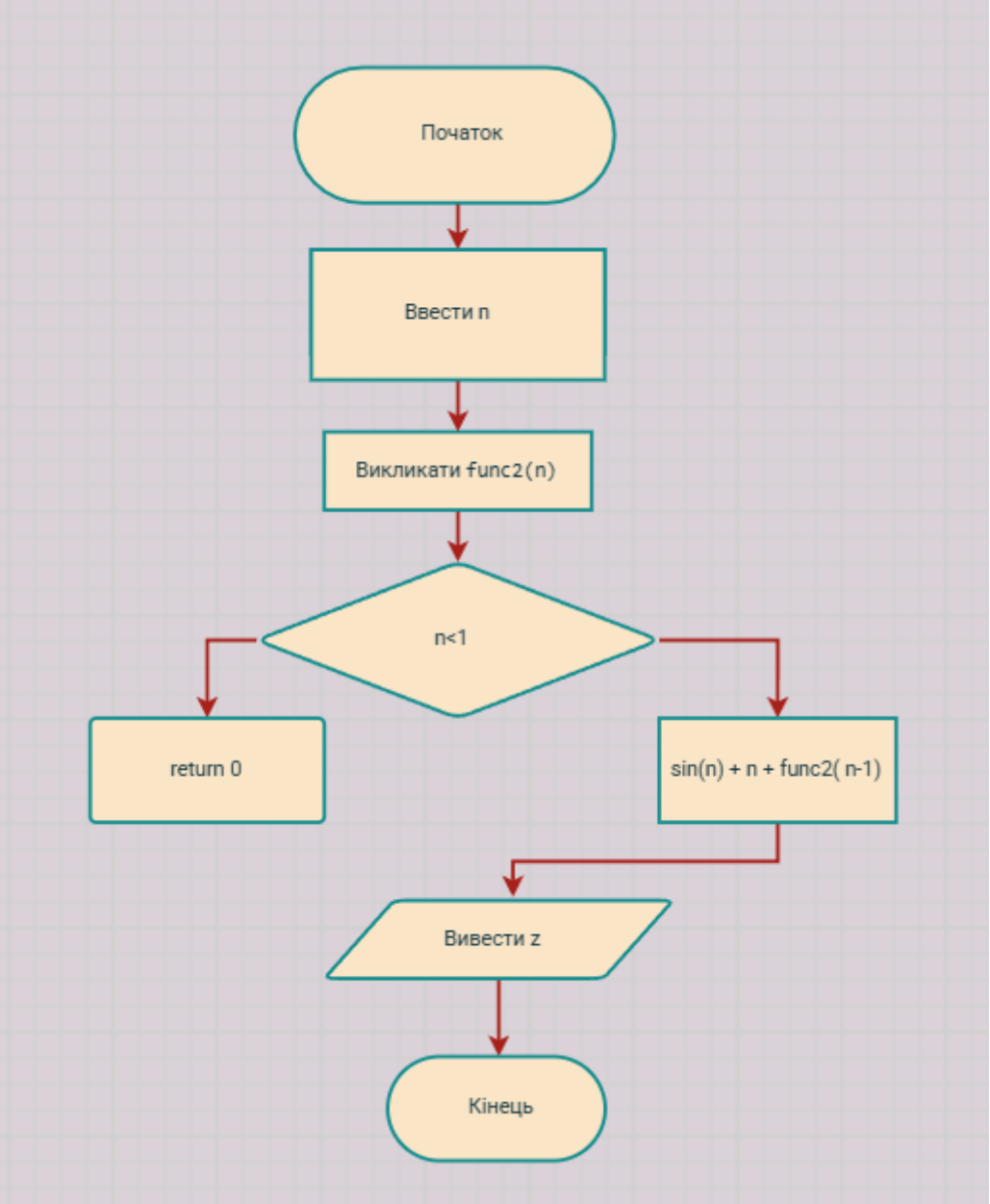
```
n = 3
Нерекурсивна функц?я = 7.89
Рекурсивна функц?я = 7.89
Чавурська Єлизавета, ЛР1, Вар?ант 20
```

Рисунок 1.5 – Вікно виконання програми 2 приклад

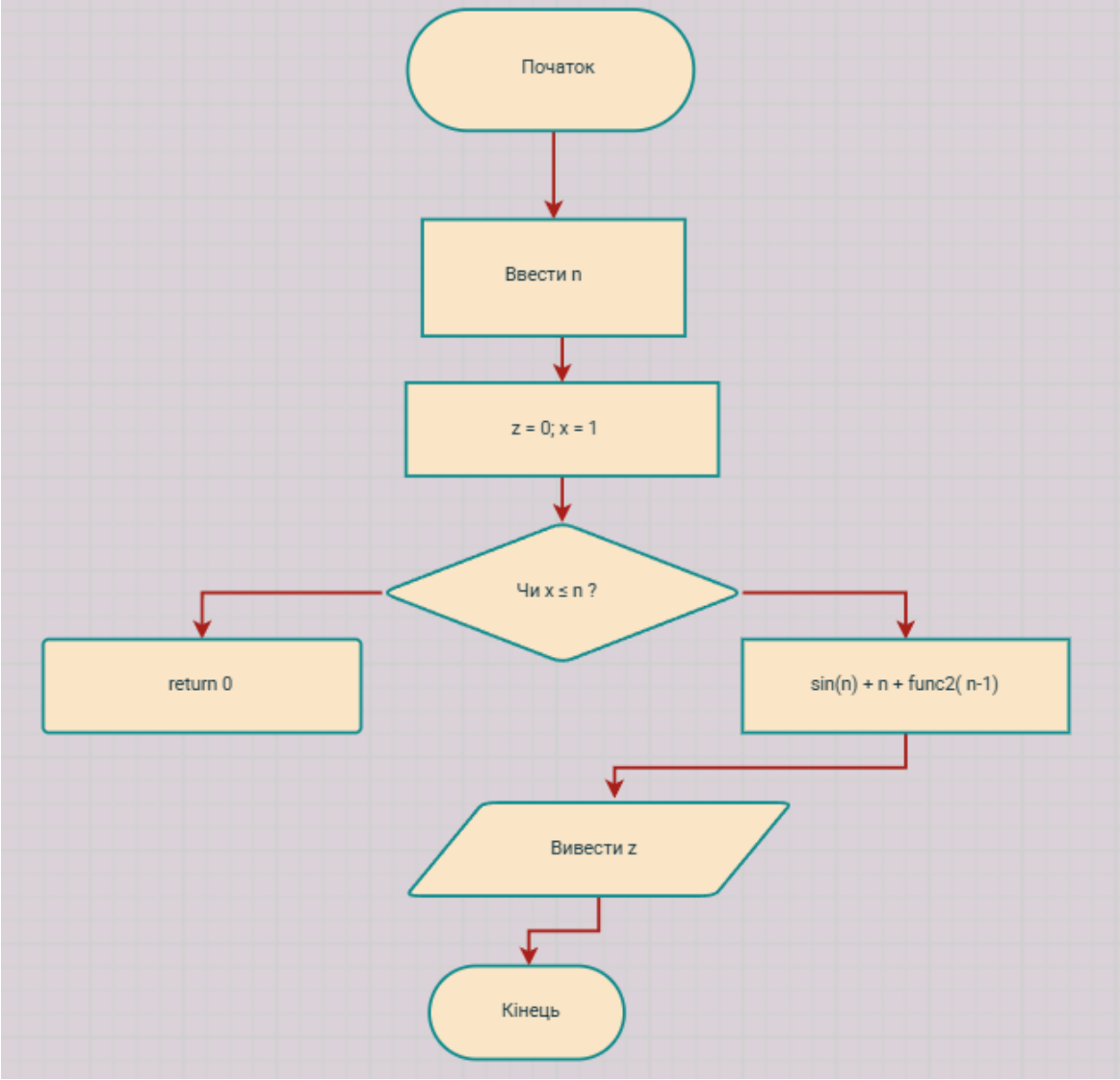
```
n = 1
Нерекурсивна функц?я = 1.84
Рекурсивна функц?я = 1.84
Чавурська Єлизавета, ЛР1, Вар?ант 20
```

Рисунок 1.6 – Вікно виконання програми 3 приклад

4. Скласти блок-схеми алгоритмів із та без використання рекурсії.




Блок-схема 1.1 – з рекурсією




Блок-схема 1.2 – без рекурсії

Висновок: на цій лабораторній роботі я навчилась створювати рекурсивну підпрограму для обчислення чисел Фібоначчі. Описувала рекурсивну функцію для визначення факторіалу числа та набувала практичних навичок роботи з рекурсивними функціями