## **МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

## **Львівський національний університет імені Івана Франка**

Факультет прикладної математики та інформатики

**ЗВІТ**

**про виконання індивідуального завдання**

**з дисципліни ”Навчальна практика”**

студентки I-го курсу

групи ПМП-12

Трей Аліси

Керівник роботи

ас. Борисюк Я.Є.

Львів – 2020

**Завдання 1.**

*Задано натуральне число k, знайти k-ту цифру последовності чисел Фібоначчі, в який виписані підряд числа Фібоначчі.*

Алгоритм:

Використовуємо бібліотеку стрічок, пишемо функцію для чисел фібоначчі. В головний програмі за допомогою неї шукаємо K.

Програма:

﻿#include <iostream>

#include <string>

using namespace std;

string fibo(int index)

{

string fibo\_string = "";

int i;

int fibo1 = 1;

int fibo2 = 1;

int fibo3 = 0;

for (i = 1; fibo\_string.length() < index; i++)

{

fibo\_string += to\_string(fibo1);

fibo1 = fibo2 + fibo3;

fibo3 = fibo2;

fibo2 = fibo1;

}

return fibo\_string;

}

int main()

{

int k;

cout << "Enter index (k): ";

cin >> k;

cout<< "Fibonacci sequence: "<< fibo(k) << endl;

cout << "K Fibo number: " << fibo(k)[k - 1];

return 0;

}

Приклад:

Enter index (k): 3

Fibonacci sequence: 112

K Fibo number: 2

**Завдання 2:**

*1. Знайти скільки днів пройшло від датиі до дати2.*

*2. Знайти на які дні тижня припадав чи буде припадати Новий рік на проміжку від year І до уеаг2.*

*3. Знайти скільки п’ятниць, 13, буде (було) в заданому році уеаг2.*

*4. Знайти в який день тижня Ви народилися (дата 1)*

*5. Задано k -ціле число та дата2. Визначити який день тижня та яка дата була k днів тому.*

Алгоритм:

Використовуємо класи. Пишемо функції для визначення дня тижня через індекс, для тривалості місяцю, для додавання і видалення одного дня з класу, для рахування кількості днів між двома датами, для пошуку індексу дня тижня, перевірки на “п'ятницю 13”, для друку. Використовуємо їх в головній програмі з додаванням простих циклів.

Програма:

#include <iostream>

#include <locale>

#include <Windows.h>

using namespace std;

enum week\_day { MONDAY = 0, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY };

class format\_date

{

public:

int year;

int month;

int day;

bool leap;

string wday\_string;

week\_day wday;

int wday\_index;

int m\_length;

void define\_wday() // визначення дня тижня через індекс

{

switch (wday\_index)

{

case 1:

wday = MONDAY;

break;

case 2:

wday = TUESDAY;

break;

case 3:

wday = WEDNESDAY;

break;

case 4:

wday = THURSDAY;

break;

case 5:

wday = FRIDAY;

break;

case 6:

wday = SATURDAY;

break;

case 7:

wday = SUNDAY;

break;

default:

break;

}

switch (wday\_index)

{

case 1:

wday\_string = "monday";

break;

case 2:

wday\_string = "tuesday";

break;

case 3:

wday\_string = "wednesday";

break;

case 4:

wday\_string = "thursday";

break;

case 5:

wday\_string = "friday";

break;

case 6:

wday\_string = "saturday";

break;

case 7:

wday\_string = "sunday";

break;

default:

break;

}

}

void d\_m\_length() //метод для тривалості місяців

{

switch (month) {

case 1:

m\_length = 31;

break;

case 2:

if (leap)

m\_length = 29;

else

m\_length = 28;

break;

case 3:

m\_length = 31;

break;

case 4:

m\_length = 30;

break;

case 5:

m\_length = 31;

break;

case 6:

m\_length = 30;

break;

case 7:

m\_length = 31;

break;

case 8:

m\_length = 31;

break;

case 9:

m\_length = 30;

break;

case 10:

m\_length = 31;

break;

case 11:

m\_length = 30;

break;

case 12:

m\_length = 31;

break;

default:

break;

}

}

void add\_day() // додає один день в клас

{

day++;

wday\_index++;

if (wday\_index > 7)

{

wday\_index = 1;

}

define\_wday();

d\_m\_length();

if (day > m\_length)

{

month++;

day = 1;

if (month > 12)

{

month = 1;

year++;

if ((year % 4 == 0) && (year % 100 != 0))

{

leap = true;

}

else

{

leap = false;

}

}

}

d\_m\_length();

}

void remove\_day() // прибирає день з класу

{

day--;

wday\_index--;

if (wday\_index < 1)

{

wday\_index = 7;

}

define\_wday();

d\_m\_length();

if (day < 1)

{

month--;

d\_m\_length();

day = m\_length;;

if (month < 1)

{

month = 12;

d\_m\_length();

year = year - 1;

if ((year % 4 == 0) && (year % 100 != 0))

{

leap = true;

}

else

{

leap = false;

}

}

}

d\_m\_length();

}

bool define\_leap() // високосний рік чи ні

{

bool result;

if ((year % 4 == 0) && (year % 100 != 0))

result = true;

else

result = false;

return result;

}

int count\_days(format\_date date2) // рахує дні між двома датами

{

int day\_count = 0;

int day\_memo = day;

int month\_memo = month;

week\_day wday\_memo = wday;

int year\_memo = year;

while ((day != date2.day) || (month != date2.month) || (year != date2.year))

{

add\_day();

day\_count++;

}

day = day\_memo;

month = month\_memo;

year = year\_memo;

wday = wday\_memo;

return day\_count;

}

void solve\_wday() // шукає індекс дня тижня

{

format\_date comparison\_date;

comparison\_date.day = 1;

comparison\_date.month = 1;

comparison\_date.year = 2000;

comparison\_date.wday\_index = 6;

comparison\_date.define\_wday();

while ((comparison\_date.day != day) || (comparison\_date.month != month) || (comparison\_date.year != year))

{

comparison\_date.add\_day();

}

comparison\_date.define\_wday();

wday\_index = comparison\_date.wday\_index;

define\_wday();

}

bool is\_friday\_13()

{ // перевірка на пт 13

bool result = false;

if ((wday == FRIDAY) and (day == 13))

result = true;

return result;

}

void print\_date()

{ //друкує

cout << day << '.' << month << '.' << year << " - " << wday\_string;

}

};

int main()

{

SetConsoleOutputCP(1251);

format\_date birthday, current, datex, datey, datez;

int year1, year2, year, friday\_13\_count = 0, removal;

cout << "Введіть дату народження: ";

cin >> birthday.day >> birthday.month >> birthday.year;

cout << "Введіть поточну дату: ";

cin >> current.day >> current.month >> current.year;

birthday.define\_leap();

cout << birthday.count\_days(current) << " днів пройшло." << endl;

cout << endl;

// 1

cout << "Введіть рік #1: ";

cin >> year1;

cout << "Введіть рік #2: ";

cin >> year2;

datex.day = 1;

datex.month = 1;

for (datex.year = year1; datex.year <= year2; datex.year++)

{

datex.solve\_wday();

datex.print\_date();

cout << endl;

}

cout << endl;

//2

cout << "Введіть рік для перевірки: ";

cin >> year;

datey.day = 1;

datey.month = 1;

datey.year = year;

datey.solve\_wday();

while (datey.year == year)

{

if (datey.is\_friday\_13())

{

friday\_13\_count++;

datey.print\_date();

cout << endl;

}

datey.add\_day();

datey.define\_wday();

}

cout << "П'ятниця 13: " << friday\_13\_count;

cout << endl;

//3

cout << endl;

cout << "Введіть дату народження: ";

cin >> birthday.day >> birthday.month >> birthday.year;

birthday.solve\_wday();

cout << birthday.wday\_string;

cout << endl;

//4

cout << endl;

cout << "Введіть дату: ";

cin >> datez.day >> datez.month >> datez.year;

cout << "Введіть кількість днів для віднімання: ";

cin >> removal;

for (int i = 0; i < removal; i++)

{

datez.remove\_day();

}

datez.print\_date();

cout << endl;

//5

system ("pause");

return 0;

}

Приклад:

Введіть дату народження: 30 01 2002

Введіть поточну дату: 20 04 2020

6656 днів пройшло.

Введіть рік #1: 2020

Введіть рік #2: 2022

1.1.2020 - wednesday

1.1.2021 - friday

1.1.2022 - saturday

Введіть рік для перевірки: 2020

13.3.2020 - friday

13.11.2020 - friday

П'ятниця 13: 2

Введіть дату народження: 30 01 2002

wednesday

Введіть дату: 20 04 2020

Введіть кількість днів для віднімання: 3

17.4.2020 - friday

**Завдання 3(1):**

*Задано масив цілих чисел А розміру n. Сформувати новий масив В, який буде містити лише ті елементи масиву А, які містяться між першим мінімальним і останнім максимальним елементами масиву А. Причому елементи в масиві В мають бути відсортовані в порядку зростання. Сортувати зразу, при формуванні масиву В. Тобто, зразу записувати елементи на правильні позиції (методом включення).*

Алгоритм:

Створюю функція для знаходження першого мінімального та максимального елементу, для виведення масиву, для перевірки на наявність числа в масиві, для збільшення розміру на 1(+1), для сортування масиву методом вставкою і додавання елемента в масив: вставляю 1 цифру в відсортований масив, щоб він все одно був посортований після того, для очищення масиву

В головній програмі ввожу стрічку, або якщо прибрати // в 95 стрічці, то можна ввести іі

самостійно

Шукаємо арифметичні прогресії та використовуємо задані раніше функції

Задаємо масив з індексами, які треба видалити

Програма:

#include <iostream>

using namespace std;

int findFirstMin(int array[], int size)

{

int min = array[0];

int position = 0;

for (int i = 1; i < size; i++)

{

if (array[i] < min)

{

min = array[i];

position = i;

}

}

return position;

}

int findLastMax(int array[], int size)

{

int max = array[0];

int position = 0;

for (int i = 1; i < size; i++)

{

if (array[i] >= max)

{

max = array[i];

position = i;

}

}

return position;

}

void print(int array[], int size)

{

for (int i = 0; i < size; i++)

cout << array[i] << ' ';

cout << endl;

}

bool check(int probe, int checked\_array[], int size)

{

bool result = false;

for (int i = 0; i < size; i++)

{

if (checked\_array[i] == probe)

{

result = true;

break;

}

}

return result;

}

void resize(int old\_array[], int size)

{

int\* new\_array = NULL;

new\_array = new int[size + 1];

for (int i = 0; i < size; i++)

new\_array[i] = old\_array[i];

new\_array[size] = 0;

old\_array = new\_array;

delete new\_array;

}

void insertionSort(int number, int array[], int size)

{

resize(array, size);

if (number <= array[0])

{

int memo1 = array[0];

int memo2;

for (int i = 1; i < size + 1; i++)

{

memo2 = array[i];

array[i] = memo1;

memo1 = memo2;

}

array[0] = number;

}

else if (number > array[size - 1])

array[size] = number;

else {

for (int i = 0; i < size; i++)

{

if ((number > array[i]) and (number <= array[i + 1]))

{

int memo1 = array[i + 1];

int memo2;

for (int j = i + 2; j < size + 1; j++)

{

memo2 = array[j];

array[j] = memo1;

memo1 = memo2;

}

array[i + 1] = number;

break;

}

}

}

}

int\* clean(int old\_array[], int check\_array[], int delta, int size)

{

int\* newArr = NULL;

newArr = new int[size - delta];

int j = 0;

for (int i = 0; i < size; i++)

{

if (!check(i, check\_array, delta))

{

newArr[j] = old\_array[i];

j++;

}

}

old\_array = newArr;

return newArr;

}

int main()

{

string input0;

cout << "Enter array: ";

//cin >> raw\_input;

input0 = "2 1 4 8 2 3 4 5 3 27 9";

int memberCount = 1;

for (char c : input0)

{

if (c == ' ')

memberCount += 1;

}

int\* input = NULL;

input = new int[memberCount];

for (int i = 0; i < memberCount; i++)

input[i] = 0;

int index = 0;

for (char c : input0)

{

if (c != ' ')

input[index] = input[index] \* 10 + (c - '0');

else

index++;

}

int first\_min = findFirstMin(input, memberCount);

int last\_max = findLastMax(input, memberCount);

print(input, memberCount);

int\* key\_array = NULL;

key\_array = new int[1];

key\_array[0] = input[0];

int current\_key\_size = 1;

for (int i = first\_min; i <= last\_max; i++)

{

if (!check(input[i], key\_array, current\_key\_size))

{

insertionSort(input[i], key\_array, current\_key\_size);

current\_key\_size++;

}

}

print(key\_array, current\_key\_size);

int step\_array[memberCount - 1];

for (int i = 0; i < memberCount - 1; i++)

step\_array[i] = input[i + 1] - input[i];

int current\_step = step\_array[0];

int current\_start = 0;

int current\_length = 1;

int delete\_index = 0;

int deleteLen = 0;

for (int i = 1; i < memberCount; i++)

{

if (step\_array[i] == current\_step)

current\_length++;

else {

if (current\_length > 2)

{

deleteLen += current\_length + 1;

}

current\_start = i;

current\_step = step\_array[i];

current\_length = 1;

}

}

int deleteArr[deleteLen];

current\_step = step\_array[0];

current\_start = 0;

current\_length = 1;

delete\_index = 0;

for (int i = 1; i < memberCount; i++)

{

if (step\_array[i] == current\_step)

current\_length++;

else {

if (current\_length > 2)

{

for (int j = 0; j <= current\_length; j++)

{

deleteArr[delete\_index] = current\_start + j;

delete\_index++;

}

}

current\_start = i;

current\_step = step\_array[i];

current\_length = 1;

}

}

input = clean(input, deleteArr, deleteLen, memberCount);

memberCount -= deleteLen;

print(input, memberCount);

delete input;

return 0;

}

Приклад:

Enter array: 2 1 4 8 2 3 4 5 3 27 9

1 2 3 4 5 8 27

2 1 4 8 3 27 9

**Завдання 4(1):**

*Вивести всі слова зі стрічки відсортовані в алфавітному порядку:*

*-функція 2 , яка розбиває стрічку на слова та повертає масив слів;*

*-функція 3, яка сортує заданий масив слів в алфавітному порядку;*

*-функція 4, яка виводить заданий масив слів на екран;*

*-функція 1, яка отримує на вхід стрічку і виводить на екран всі слова стрічки відсортовані в алфавітному порядку (використовує три інші функції);*

Алгоритм:

Створюємо функцію для визначення мінімальної довжини стрічки, для порівняння стрічок, для виводу масиву слів, фунція для сортуванням за абеткою, для парсингу і виводу відсортованої стрічки слів з масиву.

Програма:

#include <iostream>

using namespace std;

int minlen (string string1, string string2)

{

unsigned long min\_len;

if (string1.length() > string2.length())

min\_len = string2.length();

else

min\_len = string1.length();

return min\_len;

}

bool compareStr (string str1, string str2)

{

bool result = false;

for (int i = 0; i < minlen(str1, str2); i++)

{

if (str1[i] < str2[i])

{

result = false;

break;

break;

} else if (str1[i] > str2[i])

{

result = true;

break;

break;

}

}

return result;

}

void printArr (string\* array, int length)

{

for (int i = 0; i < length; i++)

{

cout << array[i] << ' ';

}

cout << endl;

}

void sortAlphabet (string\* array, int length)

{

for (int i = 0; i < length - 1; i++)

{

for (int j = 0; j < length - i - 1; j++)

{

if (compareStr(array[j], array[j + 1]))

{

swap(array[j], array[j + 1]);

}

}

}

}

int countWords (string raw)

{

int member\_count = 1;

for (char c: raw)

{

if (c == ' ')

{

member\_count++;

}

}

return member\_count;

}

string\* parse (string raw)

{

string\* words = new string[countWords(raw)];

string current = "";

int current\_word = 0;

for (char c: raw)

{

if (c == ' ')

{

words[current\_word] = current;

current.clear();

current\_word++;

} else

current += c;

words[current\_word] = current;

}

return words;

}

void print (string raw)

{

string\* words = parse(raw);

sortAlphabet(words, countWords(raw));

printArr(words, countWords(raw));

}

int main()

{

cout << "Sorted by alphabet array is: ";

print("bike count book apple programming zoom");

system("pause");

return 0;

}

Приклад:

(Початкове значення bike count book apple programming zoom )

Sorted by alphabet array is: apple bike book count programming zoom

**Завдання 4(1), 2 частина:**

*Створити масив із тих слів стрічки, які відповідають умові – перша буква слова зустрічається у ньому ще раз (ще один лише раз). Вивести масив. Видалити дані слова зі стрічки. Вивести оновлену стрічку.*

Алгоритм:

Створюємо функцію для перевірки умови - перша буква має бути в слові тільки два рази, створюємо функцію для виводу масиву слів, для лічильника слів, парсингу стрічки і для видалення елементів зі стрічки. Стрічку вводимо в програмі(можна змінити для вводу з консолі)

Програма:

#include <iostream>

using namespace std;

bool repeat (string word)

{

bool result;

int first\_count = 0;

for (char c: word)

{

if (c == word[0])

{

first\_count++;

}

}

if (first\_count == 2)

{

result = true;

}

else

{

result = false;

}

return result;

}

void displayArr (string\* array, int length)

{

for (int i = 0; i < length; i++)

{

cout << array[i] << ' ';

}

cout << endl;

}

int countWords (string raw)

{

int member\_count = 1;

for (char c: raw)

{

if (c == ' ')

{

member\_count++;

}

}

return member\_count;

}

string\* parse (string raw)

{

string\* words = new string[countWords(raw)];

string current = "";

int current\_word = 0;

for (char c: raw)

{

if (c == ' ')

{

words[current\_word] = current;

current.clear();

current\_word++;

}

else

current += c;

words[current\_word] = current;

}

return words;

}

string clean (string raw)

{

string\* words = parse(raw);

string result = "";

for (int i = 0; i < countWords(raw); i++)

{

if (repeat(words[i]))

{

if (result != "")

result += ' ';

result += words[i];

}

}

return result;

}

int main()

{

string check = "apple bamboo hi cat dog";

cout << "Initial string: " << check<<endl;

cout << "Changed string: "<< clean(check) << endl;

return 0;

}

Приклад:

Initial string: apple bamboo hi cat dog

Changed string: bamboo