**SPRAWOZDANIE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Przedmiot** | Wprowadzenie do Informatyki | **Zadanie** | 8.4 |
| **Autor** | Andrii Godlevskyi | **Grupa** | WCY23KY1S1 |
| **Temat** | Pliki i strumienie | | |

1.Treść zadania

Zmodyfikuj program z punktu 3, tak, aby łączył dane   
z 4 plików.

Metoda realizacji

Tworzymy funkcji do: wyświetlania liczb po 10 w wierszu, sortowania liczb. Odczytujemy liczby z plików i wyświetlamy ich na ekranie zanim wywołujemy funkcje sortowania liczb, dalej zapisujemy posortowane liczby do 4 pliku i wyświetlamy ich na ekranie.

1. Założenia / ograniczenia dotyczące danych:

1. Dane wejściowe

nazwa\_pliku we1, nazwa\_pliku we2, nazwa\_pliku we3 nazwa\_pliku we4 nazwa\_pliku wy – wczytywana z klawiatury

1. Dane wyjściowe

Dane z pliku we1, dane z pliku we2, - wyświetlone na ekranie.

Dane z pliku wy - zapisywane do pliku i wyświetlone na ekranie.

1. Realizacja
2. Algorytm

Изображение выглядит как текст, рукописный текст, бумага, Бумажное изделие

Автоматически созданное описание



1. Kod zródłowy

#include <stdio.h>

#include <stdlib.h>

// Funkcja do wyswietlania liczb po 10 w wierszu

void displayNumbers(int \*numbers, int count) {

for (int i = 0; i < count; i++) {

printf("%5d", numbers[i]);

if ((i + 1) % 10 == 0) {

printf("\n");

}

}

if (count % 10 != 0) {

printf("\n");

}

}

// Funkcja do sortowania liczb

void sortNumbers(int \*numbers, int count) {

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

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

if (numbers[j] > numbers[j + 1]) {

int temp = numbers[j];

numbers[j] = numbers[j + 1];

numbers[j + 1] = temp;

}

}

}

}

int main() {

//deeklaracja zmiennych

char inputFile1[256], inputFile2[256], inputFile3[256], inputFile4[256], outputFile[256];

printf("Podaj nazwe pierwszego pliku wejsciowego: ");

scanf("%s", inputFile1);

printf("Podaj nazwe drugiego pliku wejsciowego: ");

scanf("%s", inputFile2);

printf("Podaj nazwe trzeciego pliku wejsciowego: ");

scanf("%s", inputFile3);

printf("Podaj nazwe czwartego pliku wejsciowego: ");

scanf("%s", inputFile4);

printf("Podaj nazwe pliku wyjsciowego: ");

scanf("%s", outputFile);

FILE \*inFile1 = fopen(inputFile1, "r");

FILE \*inFile2 = fopen(inputFile2, "r");

FILE \*inFile3 = fopen(inputFile3, "r");

FILE \*inFile4 = fopen(inputFile4, "r");

if (inFile1 == NULL || inFile2 == NULL || inFile3 == NULL || inFile4 == NULL) {

perror("Nie mozna otworzyc jednego z plikow wejsciowych");

return 1;

}

int numbers1[1000], numbers2[1000], numbers3[1000], numbers4[1000];

int count1 = 0, count2 = 0, count3 = 0, count4 = 0;

// Wczytywanie liczb z pierwszego pliku

while (fscanf(inFile1, "%d", &numbers1[count1]) != EOF) {

count1++;

}

fclose(inFile1);

// Wczytywanie liczb z drugiego pliku

while (fscanf(inFile2, "%d", &numbers2[count2]) != EOF) {

count2++;

}

fclose(inFile2);

// Wczytywanie liczb z trzeciego pliku

while (fscanf(inFile3, "%d", &numbers3[count3]) != EOF) {

count3++;

}

fclose(inFile3);

// Wczytywanie liczb z czwartego pliku

while (fscanf(inFile4, "%d", &numbers4[count4]) != EOF) {

count4++;

}

fclose(inFile4);

// Wypisanie liczb

printf("Liczby z pliku %s:\n", inputFile1);

displayNumbers(numbers1, count1);

printf("Liczby z pliku %s:\n", inputFile2);

displayNumbers(numbers2, count2);

printf("Liczby z pliku %s:\n", inputFile3);

displayNumbers(numbers3, count3);

printf("Liczby z pliku %s:\n", inputFile4);

displayNumbers(numbers4, count4);

// Sortowanie liczb

int mergedNumbers[4000];

int i = 0, j = 0, k = 0, l = 0, m = 0;

while (i < count1 && j < count2 && k < count3 && l < count4) {

if (numbers1[i] < numbers2[j] && numbers1[i] < numbers3[k] && numbers1[i] < numbers4[l]) {

mergedNumbers[m++] = numbers1[i++];

} else if (numbers2[j] < numbers1[i] && numbers2[j] < numbers3[k] && numbers2[j] < numbers4[l]) {

mergedNumbers[m++] = numbers2[j++];

} else if (numbers3[k] < numbers1[i] && numbers3[k] < numbers2[j] && numbers3[k] < numbers4[l]) {

mergedNumbers[m++] = numbers3[k++];

} else {

mergedNumbers[m++] = numbers4[l++];

}

}

while (i < count1 && j < count2 && k < count3) {

if (numbers1[i] < numbers2[j] && numbers1[i] < numbers3[k]) {

mergedNumbers[m++] = numbers1[i++];

} else if (numbers2[j] < numbers1[i] && numbers2[j] < numbers3[k]) {

mergedNumbers[m++] = numbers2[j++];

} else {

mergedNumbers[m++] = numbers3[k++];

}

}

while (i < count1 && j < count2 && l < count4) {

if (numbers1[i] < numbers2[j] && numbers1[i] < numbers4[l]) {

mergedNumbers[m++] = numbers1[i++];

} else if (numbers2[j] < numbers1[i] && numbers2[j] < numbers4[l]) {

mergedNumbers[m++] = numbers2[j++];

} else {

mergedNumbers[m++] = numbers4[l++];

}

}

while (i < count1 && k < count3 && l < count4) {

if (numbers1[i] < numbers3[k] && numbers1[i] < numbers4[l]) {

mergedNumbers[m++] = numbers1[i++];

} else if (numbers3[k] < numbers1[i] && numbers3[k] < numbers4[l]) {

mergedNumbers[m++] = numbers3[k++];

} else {

mergedNumbers[m++] = numbers4[l++];

}

}

while (j < count2 && k < count3 && l < count4) {

if (numbers2[j] < numbers3[k] && numbers2[j] < numbers4[l]) {

mergedNumbers[m++] = numbers2[j++];

} else if (numbers3[k] < numbers2[j] && numbers3[k] < numbers4[l]) {

mergedNumbers[m++] = numbers3[k++];

} else {

mergedNumbers[m++] = numbers4[l++];

}

}

while (i < count1 && j < count2) {

if (numbers1[i] < numbers2[j]) {

mergedNumbers[m++] = numbers1[i++];

} else {

mergedNumbers[m++] = numbers2[j++];

}

}

while (i < count1 && k < count3) {

if (numbers1[i] < numbers3[k]) {

mergedNumbers[m++] = numbers1[i++];

} else {

mergedNumbers[m++] = numbers3[k++];

}

}

while (i < count1 && l < count4) {

if (numbers1[i] < numbers4[l]) {

mergedNumbers[m++] = numbers1[i++];

} else {

mergedNumbers[m++] = numbers4[l++];

}

}

while (j < count2 && k < count3) {

if (numbers2[j] < numbers3[k]) {

mergedNumbers[m++] = numbers2[j++];

} else {

mergedNumbers[m++] = numbers3[k++];

}

}

while (j < count2 && l < count4) {

if (numbers2[j] < numbers4[l]) {

mergedNumbers[m++] = numbers2[j++];

} else {

mergedNumbers[m++] = numbers4[l++];

}

}

while (k < count3 && l < count4) {

if (numbers3[k] < numbers4[l]) {

mergedNumbers[m++] = numbers3[k++];

} else {

mergedNumbers[m++] = numbers4[l++];

}

}

while (i < count1) {

mergedNumbers[m++] = numbers1[i++];

}

while (j < count2) {

mergedNumbers[m++] = numbers2[j++];

}

while (k < count3) {

mergedNumbers[m++] = numbers3[k++];

}

while (l < count4) {

mergedNumbers[m++] = numbers4[l++];

}

printf("Polaczone liczby przed sortowaniem:\n");

displayNumbers(mergedNumbers, m);

sortNumbers(mergedNumbers, m);

printf("Polaczone i posortowane liczby:\n");

displayNumbers(mergedNumbers, m);

FILE \*outFile = fopen(outputFile, "w");

if (outFile == NULL) {

perror("Nie mozna otworzyc pliku wyjściowego");

return 1;

}

for (int i = 0; i < m; i++) {

fprintf(outFile, "%5d", mergedNumbers[i]);

if ((i + 1) % 10 == 0) {

fprintf(outFile, "\n");

}

}

if (m % 10 != 0) {

fprintf(outFile, "\n");

}

fclose(outFile);

return 0;

}

1. Dane wejściowe

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, часы

Автоматически созданное описание

1. Dane wyjściowe

Изображение выглядит как текст, снимок экрана, меню, Шрифт

Автоматически созданное описание

1. Złożoność obliczeniowa algorytmu

O(1)