**SPRAWOZDANIE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Przedmiot** | Wprowadzenie do Informatyki | **Zadanie** | 9.3 |
| **Autor** | Mateusz Jasiński | **Grupa** | WCY20IY2S1 |
| **Temat** | Pliki i strumienie – pliki binarne. | | |

1. Treść

Program łączy pliki binarne zawierające posortowane wg pola int tak, aby zapisując je do trzeciego pliku były również posortowane oraz wyświetla na ekranie (dane każdego rekordu w kolejnym wierszu). Listowanie zaczyna nagłówek.

* 1. Metoda realizacji

Odczytujemy z dwóch plików rekordy do dwóch tablic, następnie tablice scalamy do jednej tablicy i wypisujemy jej elementy na ekran oraz zapisujemy je do trzeciego pliku.

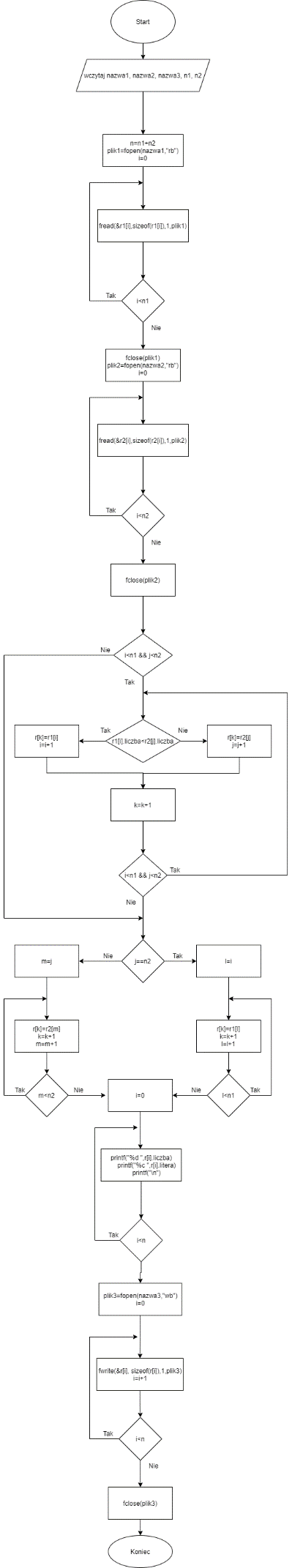
* 1. Założenia / ograniczenia dotyczące danych:
     1. Dane wejściowe

Nazwy trzech plików – wprowadzane z klawiatury

* + 1. Dane wyjściowe

n posortowanych rekordów – wyprowadzane na ekran i zapisane do trzeciego pliku.

1. Realizacja
   1. Algorytm



* 1. Kod źródłowy

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct pola {

int i;

char ch;

} pola;

pola \*wczytaj(char \*nazwa\_pliku1, char \*nazwa\_pliku2, pola \*tab, int \*n) {

FILE\* plik1=fopen(nazwa\_pliku1,"rb");

FILE\* plik2=fopen(nazwa\_pliku2,"rb");

if(!plik1 || !plik2) {

printf("Nie znaleziono pliku do wczytania!");

exit(0);

}

int i, n1=0, n2=0, x1, x2;

pola \*tab1, \*tab2, \*temp;

while(fread(&temp, sizeof(tab), 1, plik1)!='\0')

n1++;

while(fread(&temp, sizeof(tab), 1, plik2)!='\0')

n2++;

rewind(plik1);

rewind(plik2);

\*n=n1+n2;

tab=(pola\*)malloc(\*n\*sizeof(tab));

tab1=(pola\*)malloc(n1\*sizeof(tab));

tab2=(pola\*)malloc(n2\*sizeof(tab));

for(i=0;i<n1;i++)

fread(tab1+i, sizeof(tab), 1, plik1);

for(i=0;i<n2;i++)

fread(tab2+i, sizeof(tab), 1, plik2);

for(i=0,x1=0,x2=0;x1<n1||x2<n2;i++) {

if(x1<n1 && x2<n2) {

if(tab1[x1].i<tab2[x2].i) {

tab[i]=tab1[x1];

x1++;

}

else {

tab[i]=tab2[x2];

x2++;

}

}

else if(x1<n1) {

tab[i]=tab1[x1];

x1++;

}

else if(x2<n2) {

tab[i]=tab2[x2];

x2++;

}

}

fclose(plik1);

fclose(plik2);

free(tab1);

free(tab2);

return tab;

}

zapisz(char \*nazwa\_pliku, pola \*tab, int n) {

FILE\* plik=fopen(nazwa\_pliku,"wb");

if(!plik) {

printf("Nie udalo sie utworzyc pliku!");

exit(0);

}

int i;

for(i=0;i<n;i++)

fwrite(tab+i, sizeof(tab), 1, plik);

fclose(plik);

}

main() {

int n, i;

pola \*tab;

char \*nazwa\_pliku1=(char\*)calloc(50, sizeof(char\*));

char \*nazwa\_pliku2=(char\*)calloc(50, sizeof(char\*));

char \*nazwa\_pliku3=(char\*)calloc(50, sizeof(char\*));

printf("Podaj nazwe pliku1 do wczytania wartosci: ");

gets(nazwa\_pliku1);

printf("Podaj nazwe pliku2 do wczytania wartosci: ");

gets(nazwa\_pliku2);

printf("Podaj nazwe pliku do zapisania wartosci: ");

gets(nazwa\_pliku3);

tab=wczytaj(nazwa\_pliku1, nazwa\_pliku2, tab, &n);

printf("\nPosortowane liczby:\nint\tchar\n");

for(i=0;i<n;i++)

printf("%d\t%c\n",tab[i].i,tab[i].ch);

zapisz(nazwa\_pliku3, tab, n);

free(nazwa\_pliku1);

free(nazwa\_pliku2);

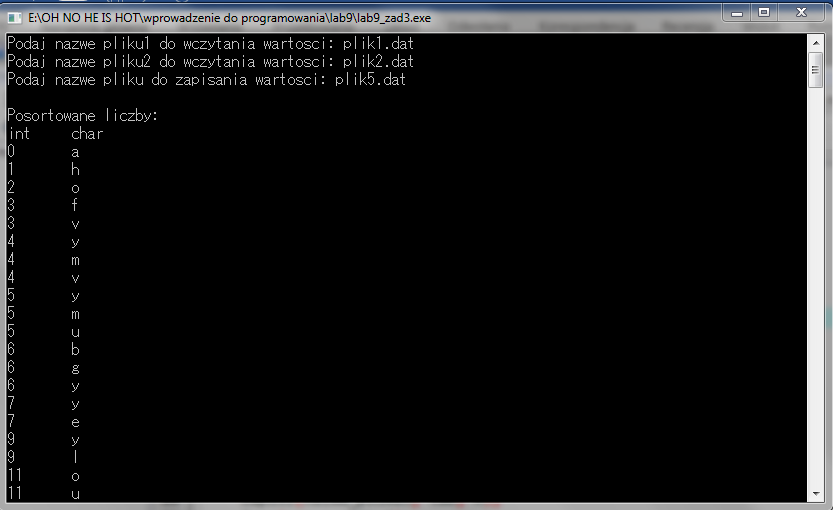
free(nazwa\_pliku3);

free(tab);

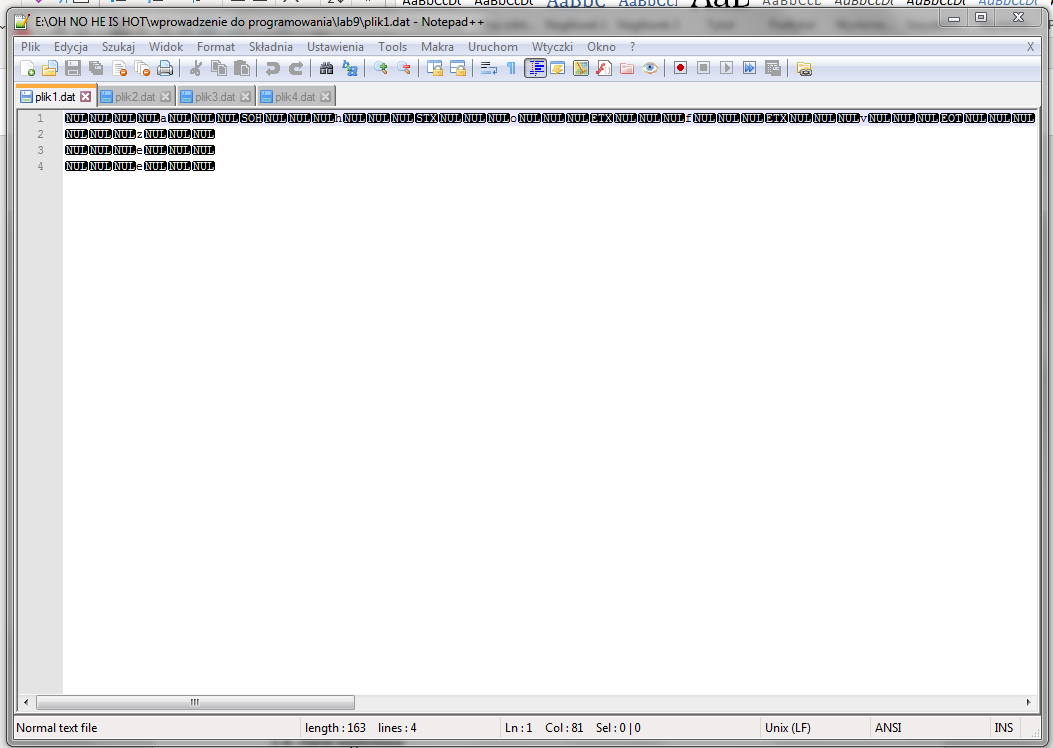
return 0;

}

* 1. Dane wejściowe



* 1. Dane wyjściowe



1. Wnioski

Złożoność obliczeniowa algorytmu:

O(n)=2+n1+n2+1+n1+n2+n+n+1=2n+2n1+2n2+4=2n+2n+4=4n+4