Systemy Operacyjne 2

Andriy Adamovych

Informatyka 3 rok lab1

**Laboratorium 10**

**Zad. 1**

**Przepisz program copy oraz przeanalizuj jego działanie.**

Przepisane i przeanalizowane. Program kopiuje zawartość pliku podanego jko pierwszy parametr do pliku, podanego jako drugi parametr.

**Zad. 2**

**Przepisz program size oraz przeanalizuj jego działanie.**

Przepisane i przeanalizowane. Program pokazuje rozmiar pliku.

**Zad. 3**

**Napisz program kopiujący zawartość pliku o nazwie podanej jako pierwszy parametr do pliku którego nazwa podana jest jako drugi parametr.**

Program:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#define MAX 512

int main(int argc, char\* argv[]) {

    char buf[MAX];

    int plik\_open, plik\_close, lbajt;

    if (argc < 3) {

        fprintf(stderr, "Za malo argumentow. Uzyj:\n");

        fprintf(stderr, "%s <plik zrodlowy> <plik docelowy>\n", argv[0]);

        exit(1);

    }

    plik\_open = open(argv[1], O\_RDONLY);

    if (plik\_open == -1) {

        perror("Blad otwarcia pliku zrodlowego.\n");

        exit(1);

    }

    plik\_close = creat(argv[2], 0640);

    if (plik\_close == -1) {

        perror("Blad utworzenia pliku docelowego.\n");

        exit(1);

    }

    while ((lbajt = read(plik\_open, buf, MAX)) > 0) {

        if (write(plik\_close, buf, lbajt) == -1) {

            perror("Blad zapisu pliku docelowego.\n");

            exit(1);

        } else {

            printf("Zapisane.\n");

        }

    }

    if (lbajt == -1) {

        perror("Blad odczytu pliku zrodlowego.\n");

        exit(1);

    }

    if (close(plik\_open) == -1 || close(plik\_close) == -1) {

        perror("Blad zamkniecia pliku.\n");

        exit(1);

    }

    exit(0);

}

Wynik:



**Zad. 4 \***

**Napisz program zmieniający kolejność znaków w każdej linii pliku o nazwie podanej jako parametr.**

Kod:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <string.h>

#define MAX 512

int main(int argc, char \*argv[])

{

    char buf[MAX];

    FILE \*plik\_otw, \*plik\_zam;

    if (argc < 2)

    {

        fprintf(stderr, "Za malo argumentow. Uzyj:\n");

        fprintf(stderr, "%s <plik zrodlowy> <plik docelowy>\n", argv[0]);

        exit(1);

    }

    plik\_otw = fopen(argv[1], "r");

    plik\_zam = fopen(argv[2], "w");

    if (plik\_otw == NULL)

    {

        perror("Blad otwarcia pliku zrodlowego.\n");

        exit(1);

    }

    if (plik\_zam == NULL)

    {

        perror("Blad otwarcia pliku docelowego.\n");

        exit(1);

    }

    char \*lines[50] = {NULL};

    int i, j, k;

    for (i = 0; fgets(buf, MAX, plik\_otw) != NULL;)

    {

        free(lines[i]);

        lines[i] = strdup(buf);

        i = (i + 1) % 50;

    }

    for (j = i;;)

    {

        if (lines[i]){

            for(k = strlen(lines[i])-1; k >= 0; k--){

                fputc(lines[i][k], plik\_zam);

            }

        }

        i = (i + 1) % 50;

        if (i == j)

            break;

    }

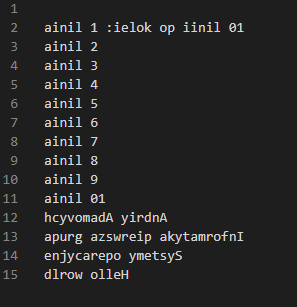
    fclose(plik\_otw);

    fclose(plik\_zam);

    exit(0);

}

Wynik:

****

**Zad. 5 \***

**Napisz procedurę kopiowania ostatnich 10 znaków, słów i ostatnich 10 linii jednego pliku do innego.**

Znaki**:**

char ch;

        fseek(plik\_otw, 0, SEEK\_END);

        int len = ftell(plik\_otw);

        fseek(plik\_otw, (len - 10), SEEK\_SET);

        while((ch = fgetc(plik\_otw)) != EOF){

            putc(ch, plik\_zam);

        }

Linie:

        char \*lines[10] = {NULL};

        int i, j;

        for(i = 0; fgets(buf, MAX, plik\_otw) != NULL;){

            free(lines[i]);

            lines[i] = strdup(buf);

            i = (i + 1) % 10;

        }

        for(j = i;;){

            if(lines[i])

                fputs(lines[i], plik\_zam);

            i = (i + 1) % 10;

            if(i == j)

                break;

        }

Słowa:

char a[10][25];

        char buf[25];

        while (fscanf(plik\_otw, "%s", buf) != EOF)

        {

            int i = 9;

            for (i; i > 0; i--)

                strcpy(a[i], a[i - 1]);

            strcpy(a[0], buf);

        }

        int i = 9;

        for (i; i >= 0; i--)

        {

            fprintf(plik\_zam, a[i]);

            putc('\n', plik\_zam);

        }

**Zad. 6**

**Napisz program do rozpoznawania czy plik o podanej nazwie jest plikiem tekstowym (plik tekstowy zawiera znaki o kodach 0-127 - można w tym celu użyć funkcji isascii.**

Kod:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <string.h>

#include <ctype.h>

#define MAX 512

int main(int argc, char\* argv[]) {

    char buf[MAX];

    FILE \*plik\_otw, \*add;

    char ch;

    if (argc < 1) {

        fprintf(stderr, "Za malo argumentow. Uzyj:\n");

        fprintf(stderr, "%s <plik zrodlowy>\n", argv[0]);

        exit(1);

    }

    plik\_otw = fopen(argv[1], "r");

    if (plik\_otw == NULL) {

        perror("Blad otwarcia pliku zrodlowego.\n");

        exit(1);

    }

    int a = 1;

    while ((ch = fgetc(plik\_otw)) != EOF){

        if(isascii(ch) == 0){

            printf("Plik %s nie jest plikiem tekstowym.\n", argv[1]);

            a = 0;

            exit(1);

        }

    }

    if (a == 1){

        printf("Plik %s jest plikiem tekstowym.\n", argv[1]);

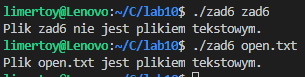
    }

    fclose(plik\_otw);

    exit(0);

}

Wynik:

****

**Zad. 7**

**Napisz program, który w pliku o nazwie podanej jako ostatni argument zapisze połączoną zawartość wszystkich plików których nazwy zostały podane w linii poleceń przed ostatnim argumentem.**

Kod:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <string.h>

#define MAX 512

int main(int argc, char \*argv[])

{

    char buf[MAX];

    FILE \*plik\_otw, \*plik\_zam;

    if (argc < 4)

    {

        fprintf(stderr, "Za malo argumentow. Uzyj:\n");

        fprintf(stderr, "%s <plik zrodlowy 1> ... <plik zrodlowy n> <plik docelowy>\n", argv[0]);

        exit(1);

    }

    int i;

    int n = argc-1;

    for(i = 1; i < n; i++){

        char ch;

        plik\_otw = fopen(argv[i], "r");

        if(i == 1)

            plik\_zam = fopen(argv[n], "w");

        else

            plik\_zam = fopen(argv[n], "a");

        if(plik\_otw == NULL || plik\_zam == NULL){

            printf("Błąd otwarcia pliku źrodłowego: %s.\n", argv[i]);

            exit(1);

        }

        if(plik\_zam == NULL){

            printf("Błąd otwarcia pliku zapisywania.\n");

            exit(1);

        }

        while((ch = fgetc(plik\_otw)) != EOF)

            putc(ch, plik\_zam);

        fclose(plik\_otw);

        fclose(plik\_zam);

    }

printf("Pliki zostały przekopiowane do pliku %s\n", argv[n]);

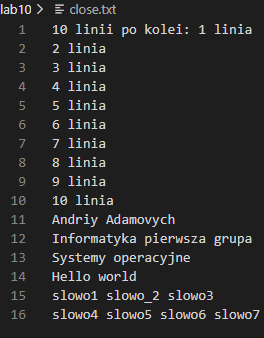
    exit(0);

}

Wynik:



Plik:



**Zad. 8**

**Napisz program liczący ile jest słów w pliku (zakładamy, że słowa składają się z małych i dużych liter alfabetu oraz cyfr i znaku podkreślenia, a wszystkie pozostałe znaki są separatorami słów).**

Kod:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <string.h>

#include <ctype.h>

#define MAX 512

int main(int argc, char\* argv[]) {

    char buf[MAX];

    FILE \*plik\_otw, \*add;

    char ch;

    if (argc < 1) {

        fprintf(stderr, "Za malo argumentow. Uzyj:\n");

        fprintf(stderr, "%s <plik zrodlowy>\n", argv[0]);

        exit(1);

    }

    plik\_otw = fopen(argv[1], "r");

    if (plik\_otw == NULL) {

        perror("Blad otwarcia pliku zrodlowego.\n");

        exit(1);

    }

    int n = 0;

    int a = 1;

    while ((ch = fgetc(plik\_otw)) != EOF){

        if(a == 1 && ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z') || (ch >= '0' && ch <= '9') || ch == '\_')){

            n++;

            a = 0;

        } else if (!((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z') || (ch >= '0' && ch <= '9') || ch == '\_')){

            a = 1;

        }

    }

    printf("Plik %s ma w sobie %d słów.\n", argv[1], n);

    fclose(plik\_otw);

    exit(0);

}

**Wynik:**

****

**Zad. 9**

**Napisz program do porównywania plików o nazwach przekazanych jako argumenty. Wynikiem działania programu ma być komunikat że pliki są identyczne, pliki różnią się od znaku nr <nr znaku> w linii <nr znaku linii> lub - gdy jeden z plików zawiera treść drugiego uzupełnioną o jakieś dodatkowe znaki - plik <nazwa> zawiera <liczba> znaków więcej niż zawartość pliku <nazwa>.**

Kod:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <string.h>

#include <ctype.h>

#define MAX 512

int main(int argc, char \*argv[])

{

    char buf[MAX];

    FILE \*plik\_otw, \*plik\_zam;

    if (argc < 2)

    {

        fprintf(stderr, "Za malo argumentow. Uzyj:\n");

        fprintf(stderr, "%s <plik pierwszy> <plik drugi>\n", argv[0]);

        exit(1);

    }

    plik\_otw = fopen(argv[1], "r");

    plik\_zam = fopen(argv[2], "r");

    if (plik\_otw == NULL)

    {

        perror("Blad otwarcia pliku zrodlowego.\n");

        exit(1);

    }

    if (plik\_zam == NULL)

    {

        perror("Blad otwarcia pliku docelowego.\n");

        exit(1);

    }

    char ch, lt;

    int i;

    int len1, len2;

    fseek(plik\_otw, 0, SEEK\_END);

    len1 = ftell(plik\_otw);

    fseek(plik\_otw, 0, SEEK\_SET);

    fseek(plik\_zam, 0, SEEK\_END);

    len2 = ftell(plik\_zam);

    fseek(plik\_zam, 0, SEEK\_SET);

    if (len1 == len2)

    {

        int a = 1;

        int n = 1;

        for (i = 0; i < len1; i++)

        {

            ch = fgetc(plik\_otw);

            lt = fgetc(plik\_zam);

            if (ch == '\n')

                a++;

            if (ch != lt)

            {

                printf("Pliki różnią się od znaku %d w linii %d\n", i + 1, a);

                n = 0;

                break;

            }

        }

        if (n == 1)

            printf("Pliki są identyczne.\n");

    }

    else if (len1 > len2)

    {

        int a = 1;

        for (i = 0; i < len1; i++)

        {

            ch = fgetc(plik\_otw);

            lt = fgetc(plik\_zam);

            if (ch == '\n')

                a++;

            if(isascii(lt)){

                if(ch != lt){

                    printf("Pliki różnią się od znaku %d w linii %d\n", i + 1, a);

                    break;

                }

            }

            if(!isascii(ch))

                len1--;

            if (lt == EOF)

            {

                printf("Plik %s zawiera %d znaków więcej niż plik %s\n", argv[1], len1 - i, argv[2]);

                break;

            }

        }

    } else if (len2 > len1){

        int a = 1;

        for (i = 0; i < len2; i++)

        {

            ch = fgetc(plik\_otw);

            lt = fgetc(plik\_zam);

            if (lt == '\n')

                a++;

            if(isascii(ch)){

                if(lt != ch){

                    printf("Pliki różnią się od znaku %d w linii %d\n", i + 1, a);

                    break;

                }

            }

            if(!isascii(lt))

                len2--;

            if (ch == EOF)

            {

                printf("Plik %s zawiera %d znaków więcej niż plik %s\n", argv[2], len2 - i, argv[1]);

                break;

            }

        }

    }

    fclose(plik\_otw);

    fclose(plik\_zam);

    exit(0);

}

Wyniki:

