

# **Dokumentacja projektu**

**Temat: Układ mnożący liczby naturalne w systemie  
siódemkowym.**

***Promotor:***

***Wykonali:***

# SPIS TREŚCI

1.	Opis wstępny projektu	3
2.	Opis wykonania	3
2.1	Sczytanie liczb z plików do tablic	3
2.2	Mnożenie liczb	3
2.3	Zapis wyniku do pliku	4
2.4	Zwolnienie przydzielonej pamięci	4
3.	Obsługa programu	5
3.1	Uruchomienie programu	5
3.2	Obsługa błędów	5
3.3	Przykład działania	5
4.	Kod źródłowy	6

# 1. Opis wstępny projektu

Celem projektu było napisanie programu w języku ANSI C wykonującego mnożenie liczb naturalnych w systemie siódmkowym. Liczby mogą być dowolnej długości(jedynym ograniczeniem jest dostępna pamięć). Liczby podawane są w systemie siódmkowym, wynik mnożenia także zapisany jest w systemie siódmkowym.

Liczby zapisane są w plikach, wynik także zapisywany jest do pliku. Program wywoływany jest z linii poleceń z trzema parametrami. Nazwy plików przekazywane są przez parametry.

# 2. Opis wykonania

## Szczytanie liczb z plików do tablic

Liczby zapisane w plikach są szczytywane do tablic. Pamięć na tablice jest alokowana w sposób dynamiczny (*funkcja malloc*). Przyjęliśmy następujące rozwiązanie: Na początku alokujemy pamięć na 50 znaków. Liczba jest szczytywana z pliku po jednym znaku, jeżeli szczytane zostanie 50 znaków pamięć przydzielona dla tablicy jest realokowana(*funkcja realloc*) i zwiększana o kolejne 50 znaków itd. Jeżeli zostanie szczytana cała liczba, wiemy już jaki jest jej rozmiar i następuje ostateczna realokacja pamięci dla tablicy. Uznaliśmy, że takie rozwiązanie będzie korzystniejsze niż realokowanie pamięci po każdym wczytaniu jednego znaku.

Wczytywanie liczb jest zrealizowane w funkcji

*char\* zpliku(char \*nazwa,unsigned long int \*i),*

która przyjmuje jako parametr nazwę pliku, a przekazuje wskaźnik do tablicy, w której została zapisana liczba oraz rozmiar tej tablicy. W funkcji tej dodatkowo sprawdzana jest poprawność formatu liczby zapisanej w pliku. Jeśli liczba ta nie jest liczbą zapisaną w systemie siódmkowym program zgłasza komunikat o błędzie i zwalnia przydzielony już obszar pamięci.

## Mnożenie liczb

Z punktu widzenia wykonania mnożenia liczb wygodniej jest jeśli w tablicy liczba jest zapisana w ten sposób, iż pod indeksem zerowym znajduje się najmniej znacząca cyfra liczby. Po szczytaniu liczby z pliku jest jednak odwrotnie, więc konieczne jest odwrócenie kolejności cyfr zapisanych w tablicy. Do wykonania tego zadania służy funkcja:

*char\* odwroc(char \*tab,unsigned long int max).*

Mnożenie liczb realizowane jest w funkcji :

*mnozenie(unsigned long int IA,unsigned long int IB,unsigned long int \*IW,char \*A,char \*B,char \*W)*

która przyjmuje jako parametry rozmiar liczby A, rozmiar liczby B i wskaźniki na tablice, w których zapisane są te liczby, natomiast zwraca wskaźnik na tablicę gdzie znajduje się wynik oraz rozmiar wyniku.

Algorytm mnożenia jest przedstawiony na poniższym przykładzie:

Mamy dwie liczby  $A=1526$  i  $B=142$

- następuje alokacja pamięci na wynik (maksymalny rozmiar wyniku jest równy sumie rozmiarów liczb  $A$  i  $B$  powiększonej o 1, w tym konkretnym przypadku wynosi 8)
- następuje alokacja pamięci na bufor, w którym przechowywane będą częściowe wyniki (maksymalny rozmiar bufora jest równy rozmiarowi liczby  $A$  +1 i w tym przypadku wynosi 5)
- pierwszy krok mnożenia wygląda następująco (mnożenie mod 7)

$$\begin{array}{r} 1562 \\ \underline{142} \\ =13454 \end{array}$$

liczba A  
liczba B  
bufor

Następnie obliczana jest suma mod 7 wyniku i bufora, po czym bufor jest zerowany i następuje drugi krok mnożenia:

$$\begin{array}{r} 1562 \\ \underline{142} \\ =10241 \end{array}$$

liczba A  
liczba B  
bufor

i ponownie sumowany jest bufor z wynikiem. Cała procedura jest powtarzana jest tyle razy ile wynosi rozmiar liczby  $B$  (w naszym przypadku 3 razy). Trzeci krok wygląda więc następująco:

$$\begin{array}{r} 1562 \\ \underline{142} \\ =01562 \end{array}$$

liczba A  
liczba B  
bufor

Końcowy wynik mnożenia wynosi: 26364

- po wykonaniu mnożenia następuje zwolnienie pamięci przydzielonej na bufor, natomiast do programu głównego przekazywany jest wskaźnik na tablicę, w której zapisany jest wynik.

### Zapis wyniku do pliku

Zapis wyniku do pliku jest zrealizowany w funkcji:

*int dopliku(char \*tab, unsigned long int n, char \*nazwa)*

Funkcja ta przyjmuje jako parametry wskaźnik na tablicę, w której znajduje się wynik, rozmiar wyniku oraz nazwę pliku do którego należy zapisać wynik.

### Zwolnienie przydzielonej pamięci

Przed zakończeniem programu zwalniana jest pamięć przydzielona na tablice, w których przechowywane były liczby i wynik (*funkcja free*).

### 3. Obsługa programu

#### Uruchomienie programu

Program uruchamiany jest z linii poleceń z trzema parametrami:

```
mnoz nazwa1 nazwa2 nazwa3
```

nazwa1 - ścieżka dostępu do pliku, w którym znajduje się pierwsza liczba

nazwa2 - ścieżka dostępu do pliku, w którym znajduje się druga liczba

nazwa3 – ścieżka dostępu do pliku, w którym zostanie zapisany wynik mnożenia

Przykładowe wywołanie:

```
mnoz c:\liczba1.txt c:\liczba2.txt d:\wynik.txt
```

#### Obsługa błędów

Jeśli wystąpi któryś z poniżej opisanych błędów, program zgłosi komunikat o błędzie, następnie zostaną wykonane niezbędne czynności umożliwiające zakończenie programu (np. zwolnienie pamięci itp.). Użytkownik ma możliwość uzyskania informacji, które mogą być pomocne przy usuwaniu źródła błędu. W programie przewidziano wystąpienie następujących błędów:

- Niepoprawny format wywołania (nieprawidłowa ilość parametrów)
- Błąd otwarcia pliku
- Nieprawidłowy format liczby zapisanej w pliku
- Błąd alokacji pamięci

#### Przykład działania

Jeżeli nie wystąpi żaden z opisanych w punkcie 3.2 błędów użytkownik zostanie poinformowany o zakończeniu mnożenia:

```
=====
PROGRAM MNOZACY LICZBY W SYSTEMIE SIODEMKOWYM
=====
Wynik mnozenia znajduje sie w pliku wynik.txt_
```

Jeśli wystąpi błąd zostanie zgłoszony odpowiedni komunikat i użytkownik ma możliwość uruchomienia pomocy:

```
=====
PROGRAM MNOZACY LICZBY W SYSTEMIE SIODEMKOWYM
=====
Błąd otwarcia pliku plik111.txt.
Wcisnij <F1> aby uzyskac wiecej informacji
Wcisnij <Esc> aby zakonczyc program
=====
POMOC
=====
Prawidlowy format wywolania programu:
nazwa_programu plik1 plik2 plik_wynikowy
plik1 - nazwa pliku(z rozszerzeniem) w ktorym znajduje sie pierwsza liczba
plik2 - nazwa pliku(z rozszerzeniem) w ktorym znajduje sie druga liczba
plik_wynikowy - nazwa pliku(z rozszerzeniem) gdzie
zostanie przeslany wynik obliczen

Liczby w plikach musza byc zapisane w systemie siodemkowym,
wynik takze jest w systemie siodemkowym
```

## 4. Kod źródłowy programu

```
#include<stdio.h>
#include<bios.h>
#include<ctype.h>
#include<conio.h>
#include<stdlib.h>
#include<alloc.h>

char* zpliku(char*, unsigned long int*);
char* odwroc(char*, unsigned long int);
char* mnozenie(unsigned long int, unsigned long int, unsigned long
int*, char*, char*, char*, char*);
int dopliku(char*, unsigned long int, char* );
int pomoc(void);

int main(int argc, char *argv[])
{
    char *liczba1, *liczba2, *wynik; //wskazniki na poszczególne tablice
    unsigned long int roz1, roz2, rozwyn, i; //rozmiary poszczegolnych tablic
    clrscr();
    textcolor(YELLOW);
    printf("=====\n\r");
    printf(" PROGRAM MNOZACY LICZBY W SYSTEMIE SIODEMKOWYM");
    printf("\r\n=====\n\r");

    if(argc!=4) //kontrola poprawnosci wywolania
    {
        textcolor(RED);
        printf("Nieprawidlowy format wywolania programu\n\r");
        pomoc();
        exit(EXIT_FAILURE);
    }
    textcolor(RED);
    if((liczba1=zpliku(argv[1], &roz1))==NULL)
        return(EXIT_FAILURE); //wczytanie z pliku pierwszej liczby, jesli
        sie nie uda wyjscie z programu
    if((liczba2=zpliku(argv[2], &roz2))==NULL)
        return(EXIT_FAILURE); //wczytanie z pliku drugiej liczby, jesli sie
        nie uda wyjscie z programu
    if((wynik=mnozenie(roz1, roz2, &rozwyn, liczba1, liczba2, wynik))==NULL)
        return(EXIT_FAILURE); //mnozenie liczb
    wynik=odwroc(wynik, rozwyn); //odwrocenie elementow w tablicy, ostatni
        zamieniany jest z pierwszym itd.

    if(!dopliku(wynik, rozwyn, argv[3])) //zapis wyniku do pliku
    {
        textcolor(GREEN);
        printf("\n\rWynik mnozenia znajduje sie w pliku %s", argv[3]);
    }
    if(liczba1) free(liczba1); //zwolnienie pamieci tablicy przechowujacej
        pierwsza liczbe
    if(liczba2) free(liczba2); //zwolnienie pamieci tablicy przechowujacej
        druga liczbe
    if(wynik) free(wynik); //zwolnienie pamieci tablicy przechowujacej wynik
    getch(); //zatrzymanie ekranu
    return(0);
}

char* zpliku(char *nazwa, unsigned long int *i) //wczytywanie liczb z pliku do
        tablicy
{
    unsigned long int max=0;
    FILE *fp;
    int c;
    char *tab; //wskaznik na tablice
    *i=0; //licznik wczytanych znakow
```

```

tab=(char*)malloc(50*sizeof(char));//alokacja pamieci dla tablicy,
                                poczatkowo pamiec jest przydzielona na 50 znakow
if(!tab)
{
    cprintf("\n\rB\Ad alokacji pamieci.");
    return(NULL); //jesli alokacja sie nie powiedzie wyjscie z funkcji
}
if((fp=fopen(nazwa,"r"))==NULL)//otwarcie pliku do odczytu
{
    cprintf("\n\rB\Ad otwarcia pliku %s.",nazwa);
    pomoc();
    return(NULL); //jesli sie nie udalo wyjscie z funkcji
}
else
{
    while((c=fgetc(fp))!=EOF) //pobranie znaku z pliku
    {
        if(!isdigit(c)|| (c-'0')>6) //kontrola poprawnosci formatu
        {
            free(tab);
            cprintf("\n\rLiczba zapisana w pliku %s nie jest liczba si~demkowa.
",nazwa);
            pomoc();
            return(NULL);
        }
        if(*i==max)
        {
            tab=(char*)realloc(tab,(*i+50)*sizeof(char));//jesli wczytano juz
                50 znakow jest alokowana pamiec na kolejne 50 znakow
            if(!tab)
            {
                cprintf("\r\nB\Ad alokacji pamieci.");
                return(NULL);
            }
            max+=50;
        }
        tab[*i]=c-'0';(*i)++;//zapis liczby odpowiadajacej danemu znakowi do
                tablicy
    }
    tab=(char*)realloc(tab,(*i)*sizeof(char));//koncowa realokacja pamieci,
                w tym momencie znany jest juz rozmiar tablicy
    fclose(fp);//zamkniecie pliku
}
tab=odwroc(tab,*i);//odwrocenie kolejnosci liczb w tablicy
                //przy wykonywaniu mnozenia korzystniej jest jesli
                //pierwszy element tablicy jest najmniej znaczaca cyfra
liczby
return(tab);//funkcja zwraca wskaznik do tablicy
}

char* odwroc(char *tab,unsigned long int max)//odwrocenie kolejnosci liczb w
                tablicy
{
    char *pom=tab; //wskaznik na poczatek tablicy
    char *kon=&tab[max-1],c; //wskaznik na koniec tablicy
    while(pom<kon) //dopuki wskazniki sie nie mina
    {
        c=*pom; //zamiana elementow
        *pom=*kon;
        *kon=c;
        ++pom;--kon;
    }
    return(tab);
}

char* mnozenie(unsigned long int lA,unsigned long int lB,unsigned long int
*lW,char *A,char *B,char *W)
{

```

```

char *buf,przenies,nadmiar,temp;
unsigned long int i,j,k,m;

(*lW)=lA+lB+1;//maksymalny rozmiar wyniku=rozmiar 1 liczby +rozmiar 2
                liczby + 1
W=(char*)calloc((*lW),sizeof(char));//alokacja pamieci na wynik
if(!W)
{
    fprintf("\n\rBład alokacji pamieci.");
    return(NULL); //jesli alokacja sie nie powiodla wyjscie z funkcji
}
buf=(char*)calloc((lA+1),sizeof(char));//alokacja pamieci na bufor
                wykorzystywany podczas mnozenia
if(!buf)
{
    fprintf("\n\rBład alokacji pamieci.");
    return(NULL); //jesli alokacja sie nie powiodla wyjscie z funkcji
}

//mnozenie
k=0;//przesuniecie bufora wzgledem wyniku
for(i=0;i<lB;i++)//dopuki i mniejsze niz rozmiar drugiej liczby
{
    for(j=0;j<(lA+1);j++) buf[j]=0; //wyzerowanie bufora
    przenies=0;//pocztkowo przeniesienie rowne zero
    for(j=0;j<lA;j++)//dopuki j mniejsze od rozmiaru bufora
    {
        if(B[i]==0) break;//jesli cyfra drugiej liczby jest rowna zero to nie
                mnoz tylko ja przeskocz
        buf[j]=(A[j]*B[i]+przenies)%7;//zapisywane na aktualnej pozycji w
                buforze
        przenies=(A[j]*B[i]+przenies)/7;//przeniesienie do nastepnej pozycji
    }
    buf[j]=przenies;//dopisz przeniesienie koncowe

    //sumowanie bufora z wynikiem
    nadmiar=0;//przeniesienie do nastepnej pozycji
    for(m=0+k;m<(*lW);m++) //dopuki m mniejsze od rozmiaru wyniku
    {
        if((m-k)<(lA+1)) //jesli istnieje pozycja m w buforze to dodaj
        {
            temp=W[m];
            W[m]=(W[m]+buf[m-k]+nadmiar)%7;
            nadmiar=(temp+buf[m-k]+nadmiar)/7;
        }
        else //jesli m wieksze niz rozmiar bufora to przepisz dalsza
                czesc wyniku
        {
            W[m]=(W[m]+nadmiar)%7;
            nadmiar=(W[m]+nadmiar)/7;
        }
    }

    k++;// zwiekszenie przesuniecie
}
if(buf) free(buf);//zwolnienie pamieci bufora
return(W);//funkcja zwraca wskaznik do tablicy zawierajacej wynik
}

int dopliku(char *tab,unsigned long int n,char *nazwa)//zapis do pliku
{
    unsigned long int i;
    FILE *fp;
    char c;
    if((fp=fopen(nazwa,"wb"))==NULL) //otwarcie pliku do zapisu
    {
        fprintf("\n\rBlad otwarcia pliku %s",nazwa);
        pomoc();
    }
}

```

```

    return(1); //jesli wystapil blad wyjscie z funkcji
}
for(i=0;i<n;i++)
{
    switch(tab[i]) //zamiana liczby na odpowiedni znak
    {
        case 0: c='0';break;
        case 1: c='1';break;
        case 2: c='2';break;
        case 3: c='3';break;
        case 4: c='4';break;
        case 5: c='5';break;
        case 6: c='6';break;
    }
    putc(c,fp);//zapis znaku do pliku

}
return(0);

}

int pomoc(void) //pomoc dla uzytkownika
{
    int k;
    printf("\n\nWcisnij <F1> aby uzyskac wiecej informacji\n");
    printf("Wcisnij <Esc> aby zakonczyc program");
    while(1)
    {
        k=bioskey(0); //czytanie znaku
        if(k==0x3b00) break;//jesli podano F1 wyjdz z petli
        if(k==0x11b) return(1);//jesli podano <Esc> wyjdz z funkcji
    }
    textcolor(LIGHTBLUE);
    cprintf("\n\n\r===== \n\r");
    cprintf("                POMOC");
    cprintf("\n\n\r===== \n\r");
    printf("\nPrawidlowy format wywolania programu:  \n");
    printf("nazwa_programu plik1 plik2 plik_wynikowy \n");
    printf("plik1 - nazwa pliku(z rozszerzeniem) w ktorym znajduje sie
           pierwsza liczba\n");
    printf("plik2 - nazwa pliku(z rozszerzeniem) w ktorym znajduje sie druga
           liczba\n");
    printf("plik_wynikowy - nazwa pliku(z rozszerzeniem) gdzie \n zostanie
           przeslany wynik obliczen \n\n ");
    printf("\n Liczby w plikach musza byc zapisane w systemie siodemkowym,
\n");
    printf("wynik takze jest w systemie siodemkowym\n\n");
    return(0);

}

```