Учреждение Образования «Белорусский Государственный Университет Информатики и Радиоэлектроники»

Отчет

По лабораторной работе №5

Структуры

Выполнил студент

Группы 252005

Волчецкий Александр Михайлович

Минск 2013

**Условие**

**5. 9.** Разработать функцию вычисления *n*!, при *n* > 100. Для хранения

многоразрядного числа использовать динамический

двунаправленный список.

**Листинг**

#include <stdio.h>

#include <malloc.h>

#include <string.h>

#include <Windows.h>

#include <locale.h>

typedef struct item

{

int digit;

struct item \*next;

struct item \*prev;

} Item;

typedef struct mnumber

{

Item \*head;

Item \*tail;

int n;

} MNumber;

MNumber CreateMNumber(char \*initStr);

void AddDigit(MNumber \*number, int digit);

void PrintMNumber(MNumber number);

MNumber MultiMNumber(MNumber n1, MNumber n2);

MNumber SumMNumber(MNumber result, MNumber factor);

void main(void)

{

setlocale( LC\_ALL,"Russian" );

int n = 0;

char str[10];

printf("Введите n=");

scanf("%d", &n);

MNumber result = CreateMNumber("1");

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

{

MNumber factor = CreateMNumber(itoa (i, str, 10));

result = MultiMNumber(result, factor);

}

printf("\t%d!=",n);

PrintMNumber(result);

printf("\n");

system("pause");

}

MNumber CreateMNumber(char initStr[])

{

MNumber number = {NULL, NULL, 0};

for (int n = strlen(initStr)-1; n >= 0; n--)

AddDigit(&number, initStr[n]-'0');

return number;

}

void AddDigit(MNumber \*number, int digit)

{

Item \*p = (Item \*)malloc(sizeof(Item));

p->digit = digit;

p->next = p->prev = NULL;

if (number->head == NULL)

number->head = number->tail = p;

else

{

number->tail->next = p;

p->prev = number->tail;

number->tail = p;

}

number->n++;

}

MNumber MultiMNumber(MNumber result, MNumber factor)

{

MNumber multi = CreateMNumber("");

MNumber summand = CreateMNumber("");

Item \*p\_result = result.head, \*p\_factor = factor.head;

int digit, pos = 0, zeros = 0, value\_result, value\_factor;

while (p\_factor || p\_result)

{

if (p\_factor)

{

value\_factor = p\_factor->digit;

p\_factor = p\_factor->next;

}

else

value\_factor = 0;

p\_result = result.head;

multi = CreateMNumber("");

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

AddDigit(&multi, 0);

zeros++;

while (p\_result)

{

if (p\_result)

{

value\_result = p\_result->digit;

p\_result = p\_result->next;

}

else

value\_result = 0;

digit = (value\_result \* value\_factor + pos) % 10;

pos = (value\_result \* value\_factor + pos) / 10;

AddDigit(&multi, digit);

}

if (pos)

AddDigit(&multi, pos);

summand = SumMNumber(multi, summand);

}

return summand;

}

MNumber SumMNumber(MNumber multi, MNumber summand)

{

MNumber sum = CreateMNumber("");

Item \*p\_multi = multi.head, \*p\_summand = summand.head;

int digit, pos = 0, value\_multi, value\_summand;

while (p\_multi || p\_summand)

{

if (p\_multi)

{

value\_multi = p\_multi->digit;

p\_multi = p\_multi->next;

}

else

value\_multi = 0;

if (p\_summand)

{

value\_summand = p\_summand->digit;

p\_summand = p\_summand->next;

}

else

value\_summand = 0;

digit = (value\_multi + value\_summand + pos) % 10;

pos = (value\_multi + value\_summand + pos) / 10;

AddDigit(&sum, digit);

}

if (pos)

AddDigit(&sum, pos);

return sum;

}

void PrintMNumber(MNumber number)

{

Item \*p = number.tail;

while (p)

{

printf("%d", p->digit);

p = p->prev;

}

}

**Скриншоты, демонстрирующие работу программы**



