Лабораторная работа №11.

**Тема: Структуры и объединения в языке С**

Задание №1.

**Постановка задачи:** Создать некоторую структуру с указателем на некоторую функцию в качестве поля. Вызвать эту функцию через имя переменной этой структуры и поле указателя на функцию

**Список идентификаторов:**

| **Переменная** | **Смысл** | **Тип** |
| --- | --- | --- |
| **r, n** | Числа для умножения | **int** |
| **pr** | Указатель на функцию | **int** |
| **obj** | экземпляр структуры | **struct mif** |
| **mif** | структура | **struct** |
| **prov** | Функция произведения | **int** |

**Код программы:**

#include <stdio.h>

#include <stdlib.h>

int prov(int r, int n)

{

return r \* n;

}

int main()

{

struct mif {

int(\*pr)(int r, int n);

} obj;

obj.pr = prov;

int r, n;

printf("You need to write two numbers = ");

scanf("%d%d",&n, &r);

printf("Result = %d", obj.pr(r,n));

getchar();

return 0;

}

**Результат:**



Задание №2.

**Постановка задачи:** Создать структуру для вектора в 3-х мерном пространстве. Реализовать и использовать в своей программе следующие операции над векторами:

* скалярное умножение векторов;
* векторное произведение;
* модуль вектора;
* распечатка вектора в консоли.

В структуре вектора указать имя вектора в качестве отдельного поля этой структуры.

**Список идентификаторов:**

| **Переменная** | **Смысл** | **Тип** |
| --- | --- | --- |
| **obj1, obj2** | экземпляр структуры | **struct vector** |
| **vector** | структура | **struct** |
| **x, y, z** | координаты векторов | **int** |
| **m, n, k** | промежуточные переменные для векторного произведения | **int** |
| **name** | имя вектора | **char** |

**Код программы:**

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <math.h>

int main()

{

char \*locale = setlocale(LC\_ALL, "");

struct vector

{

int x, y, z;

char name;

};

struct vector obj1 = { 1,8,4, {'s'} };

struct vector obj2 = { 7,4,1, {'m'} };

printf("Вектор %c = {%1d, %1d, %1d}", obj1.name, obj1.x, obj1.y, obj1.z);

printf("\nВектор %c = {%1d, %1d, %1d}", obj2.name, obj2.x, obj2.y, obj2.z);

printf("\nСкалярное произведение = %1d", obj1.x\*obj2.x + obj1.y\*obj2.y + obj1.z\*obj2.z);

int m = obj1.y\*obj2.z - obj1.z\*obj2.y,

n = -(obj1.x\*obj2.z - obj1.z\*obj2.x),

k = obj1.x\*obj2.y - obj1.y\*obj2.x;

printf("\nВекторное произведение = {%1d, %1d, %1d}", m, n, k);

printf("\nМодуль вектора %c = %.2f", obj1.name, sqrt(obj1.x\*obj1.x + obj1.y\*obj1.y + obj1.z\*obj1.z));

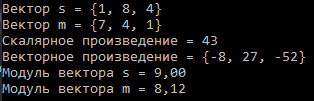
printf("\nМодуль вектора %c = %.2f", obj2.name, sqrt(obj2.x\*obj2.x + obj2.y\*obj2.y + obj2.z\*obj2.z));

getchar();

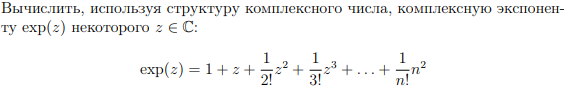
return 0;

}

**Результат:**

****

Задание №3.

**Постановка задачи:я  
**

**Список идентификаторов:**

| **Переменная** | **Смысл** | **Тип** |
| --- | --- | --- |
| **c** | экземпляр структуры | **struct complex** |
| **complex** | структура | **struct** |
| **a, b** | действительная и мнимая часть комплексного числа | **float** |
| **i** | Параметр цикла | **int** |
| **fact** | Функция вычисления факториала | **int** |
| **n** | Граница | **int** |

**Код программы:**

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <time.h>

int fact(int n)

{

if (n <= 1)

return 1;

else

{

return n \* fact(n - 1);

}

}

int main()

{

srand(time(NULL));

struct complex

{

float a, b;

}c;

int n;

float a=1, b=1;

c.a = rand() % 5;

c.b = rand() % 5;

printf("You need to write a number = ");

scanf("%d", &n);

printf("\z = %.2f + %.2f \* i ", c.a, c.b);

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

{

a += pow(c.a, i) / fact(i);

b += pow(c.b, i) / fact(i);

}

printf("\nexp(z): %.2f + %.2f \* i ", a, b);

getchar();

return 0;

}

**Результат:**



Задание №4.

**Постановка задачи:** Реализовать в виде структур двунаправленный связный список и совершить отдельно его обход в прямом и обратном направлениях с распечаткой значений каждого элемента списка.

**Список идентификаторов:**

| **Переменная** | **Смысл** | **Тип** |
| --- | --- | --- |
| **Lists** | Структура | **struct** |
| **l** | Значение символьного поля | **char** |
| **i** | Параметр цикла | **int** |
| **next/previous** | Следующий/предыдущий экземпляр структуры | **struct Lists** |
| **x, y** | Указатель на текущий/следующий экземпляр структуры | **struct Lists** |
| **start** | Указатель на начало списка | **struct Lists** |

**Код программы:**

include <stdio.h>

#include <stdlib.h>

#include <malloc.h>

typedef struct Lists

{

char l;

struct Lists \*next;

struct Lists \*previous;

};

int main()

{

struct Lists \*start = malloc(sizeof(struct Lists));

start->l = 'A';

start->previous = 0;

struct Lists \*x, \*y;

x = start;

for (int i = 1; i<5; i++)

{

y = malloc(sizeof(struct Lists));

y->l = x->l + 1;

x->next=y;

y->previous = x;

x = y;

}

x ->next = 0;

x = start;

while (x != 0)

{

printf("%2c", x->l);

x = x->next;

}

printf("\n");

x = start->next->next->next->next;

while (x != 0)

{

printf("%2c", x->l);

x = x->previous;

}

free(start ->next ->next ->next->next);

free(start->next->next->next);

free(start->next->next);

free(start->next);

free(start);

return 0;

}

**Результат:**

