Лабораторная работа № 3

Структуры. Объединения. Перечисления.

Задание 1.1

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.

Математическая модель:

-

Список идентификаторов:

Имя	**Тип**	**Назначение**
MyObj	SomeStruct{&SomeFunc}	Объект структуры SomeStruct.

Код программы:

```c
#include <stdio.h>


int
SomeFunc(int a, int b)
{
    printf("%d\n",a+b );
    return  a+b;
}

int
main()
{
    struct SomeStruct{
        int (*func)(int, int);
    };

    struct SomeStruct MyObj = {
            &SomeFunc
    };

    MyObj.func(1, 2);

    MyObj.func(10, 12);

    MyObj.func(0x0a, 0x12);
    return 0;
}
```

Результаты выполненной работы:

```
3
22
28
```

Задание 1.2

Постановка задачи:

Создать структуру для вектора в 3-х мерном пространстве. Реализовать и использховать 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.

Математическая модель:

-

Список идентификаторов:

Имя	**Тип**	**Назначение**
a	Vector	Первый вектор для демонстрации работы программы.
b	Vector	Второй вектор для демонстрации работы программы.
d	Vector	Скалярное произведение векторов A и B.
e	Vector	Произведение векторов D и B.

Код программы:

```c
#include <stdio.h>
#include <math.h>
#include <string.h>

typedef struct{
    double x;
    double y;
    double z;
    char name[20];
} Vector;

Vector VecProduct(Vector, Vector);
Vector VecMult(Vector, Vector);
double VecLenght(Vector);
void VecPrint(Vector);


int
main()
{
    Vector a = {
        12, 1, -0.9, "Alpha"
    };
    Vector b = {
        -2, 0.10, 18, "Beta"
    };
    Vector d = VecProduct(a, b);
    VecPrint(d);

    Vector e = VecMult(d, b);
    VecPrint(e);
    return 0;
}
```

```
34
35 Vector VecProduct(Vector a, Vector b)
36 {
37     Vector c;
38     c.x = a.x*b.x;
39     c.y = a.y*b.y;
40     c.z = a.z*b.z;
41     strcpy(c.name, "product");
42     return c;
43 }
44
45 Vector VecMult(Vector a, Vector b)
46 {
47     Vector c;
48     strcpy(c.name, "mult");
49     c.x = a.y*b.z - a.z*b.y;
50     c.y = a.z*b.x - a.x*b.z;
51     c.z = a.x*b.y - a.y*b.x;
52     return c;
53 }
54
55 double VecLenght(Vector c){
56     return sqrt(c.x*c.x + c.y*c.y + c.z*c.z);
57 }
58
59 void VecPrint(Vector a){
60     printf("Name:%s\n\tLen = %lf\n\tx = %lf\n\ty = %lf\n\
   tz = %lf\n", a.name, VecLenght(a), a.x, a.y, a.z);
61 }
```

Результаты выполненной работы:

```
Name:product
        Len = 28.956001
        x = -24.000000
        y = 0.100000
        z = -16.200000
Name:mult
        Len = 464.417804
        x = 3.420000
        y = 464.400000
        z = -2.200000
```

Задание 1.3

Постановка задачи:

Вычислить, используя структуру комплексного числа, комплексную экспоненту exp(z) некоторого z ∈ C:

Математическая модель:

$$\exp(z)=1+z+\frac{1}{2!}z^2\frac{1}{3!}z^3+...+\frac{1}{n!}z^n$$

Список идентификаторов:

Имя	**Тип**	**Назначение**
z	compl	Комплексное число для нахождения от него экспоненты.
y	compl	Результат нахождения экспоненты от z.

Код программы:

```c
#include <stdio.h>
#include <math.h>

struct{
    double a;
    double i;
    // c = a + i * b;
}typedef compl;

compl Exp(compl, int);
unsigned factorial(int);
void print_complex(compl z);

int
main()
{
    compl z = {4, 2};
    compl y = Exp(z, 13);
    print_complex(y);
    return 0;
}

unsigned
factorial(int a)
{
    unsigned c = 1;
    for (int i = 1; i <= a; i++)
        c *= i;
    return c;
}
```

```c
compl
Exp(compl z, int n)
{
    compl x = {1, 1};
    for (int i = 1; i <= n; i++)
    {
        x.a += 1.0/factorial(i)*pow(z.a, i);
        x.i += 1.0/factorial(i)*pow(z.i, i);
    }
    return x;
}

void
print_complex(compl z) {
    printf("%f + %fi\n", z.a, z.i);
}
```

Результаты выполненной работы:

```
54.617940 + 7.389059i
```

Задание 1.4

Постановка задачи:

Используя так называемые "битовые" поля в структуре С, создать экономную структуру в оперативной памяти для заполнения даты некоторого события, например даты рождения человека. Ссылки на описание битовых полей

Математическая модель:

-

Список идентификаторов:

Имя	**Тип**	**Назначение**
bday	DATE	Объект структуры DATE.

Код программы:

```c
#include <stdio.h>

struct{
    char name[32];
    unsigned year : 12;
    unsigned month : 4;
    unsigned day : 5;
}typedef DATE;

void print_date(DATE);

int
main()
{
    DATE bday = {
        "Birthday", 2005, 4, 28
    };
    print_date(bday);
    printf("%zu\n", sizeof(bday));
    return 0;
}

void print_date(DATE a)
{
    printf("Event:%s\n\t%d-%d-%d\n",a.name, a.day, a.month, a.year);
}
```

Результат выполненной работы:

```
Event:Birthday
        28-4-2005
36
```

Задание 1.5

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.

Математическая модель:

-

Список идентификаторов:

Имя	**Тип**	**Назначение**
a1	dlist_elem	Первый элемент двунаправленного связного списка.
a2	dlist_elem	Второй элемент двунаправленного связного списка.
a3	dlist_elem	Третий элемент двунаправленного связного списка.
a4	dlist_elem	Четвёртый элемент двунаправленного связного списка.
a5	dlist_elem	Пятый элемент двунаправленного связного списка.
a6	dlist_elem	Шестой элемент двунаправленного связного списка.

Код программы:

```c
#include <stdio.h>
#include <stdlib.h>


struct dlist_elem{
    int feild;
    struct dlist_elem *prev;
    struct dlist_elem *next;
};

void backward_print_dlist(int count, struct dlist_elem c);
void print_dlist(int count, struct dlist_elem c);

int
main()
{
    struct dlist_elem a1 = {10, NULL, NULL};
    struct dlist_elem a2 = {11, &a1, NULL};
    a1.next = &a2;

    struct dlist_elem a3 = {12, &a2, NULL};
    a2.next = &a3;

    struct dlist_elem a4 = {13, &a3, NULL};
    a3.next = &a4;

    struct dlist_elem a5 = {14, &a4, NULL};
    a4.next = &a5;

    struct dlist_elem a6 = {15, &a5, NULL};
    a5.next = &a6;

    backward_print_dlist(5, a5);
    print_dlist(5, a1);
}

void
print_dlist(int count, struct dlist_elem c)
{
    putchar('\n');
    struct dlist_elem *iter = &c;
    for(int i = 1; i <= count; i++)
    {
        printf("%d)Elem: %p\n\tField: %d\n\tNext: %p\n\tPrev: %p\n", i,iter, iter->feild, iter->next, iter->prev);
        iter = iter->next;
    }
}

```

```c
void
backward_print_dlist(int count, struct dlist_elem c)
{
    putchar('\n');
    struct dlist_elem *iter = &c;
    for(int i = count; i > 0; i--)
    {
        printf("%d)Elem: %p\n\tField: %d\n\tNext: %p\n\tPrev: %p\n", i,iter, iter->feild, iter->next, iter->prev);
        iter = iter->prev;
    }
}
```

Результат выполненной работы:

```
5)Elem: 0x7ffe05b66660
        Field: 14
        Next: 0x7ffe05b66720
        Prev: 0x7ffe05b666e0
4)Elem: 0x7ffe05b666e0
        Field: 13
        Next: 0x7ffe05b66700
        Prev: 0x7ffe05b666c0
3)Elem: 0x7ffe05b666c0
        Field: 12
        Next: 0x7ffe05b666e0
        Prev: 0x7ffe05b666a0
2)Elem: 0x7ffe05b666a0
        Field: 11
        Next: 0x7ffe05b666c0
        Prev: 0x7ffe05b66680
1)Elem: 0x7ffe05b66680
        Field: 10
        Next: 0x7ffe05b666a0
        Prev: (nil)

1)Elem: 0x7ffe05b66660
        Field: 10
        Next: 0x7ffe05b666a0
        Prev: (nil)
2)Elem: 0x7ffe05b666a0
        Field: 11
        Next: 0x7ffe05b666c0
        Prev: 0x7ffe05b66680
3)Elem: 0x7ffe05b666c0
        Field: 12
        Next: 0x7ffe05b666e0
        Prev: 0x7ffe05b666a0
4)Elem: 0x7ffe05b666e0
        Field: 13
        Next: 0x7ffe05b66700
        Prev: 0x7ffe05b666c0
5)Elem: 0x7ffe05b66700
        Field: 14
        Next: 0x7ffe05b66720
        Prev: 0x7ffe05b666e0
```

Задание 2.1

Постановка задачи:

Напишите программу, которая использует указатель на некоторое объединение union.

Математическая модель:

-

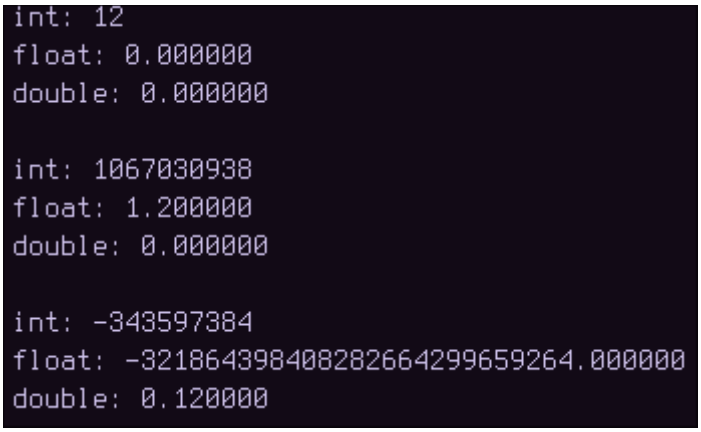
Список идентификаторов:

Имя	**Тип**	**Назначение**
u	MyUnion	Некоторое объединение.
pu	*MyUnion	Указатель на некоторое объединение.

Код программы:

```c
#include <stdio.h>

union MyUnion{
    int     num;
    float   fnum;
    double  dnum;
};

void
print_all_members(union MyUnion u)
{
    printf("int: %d\nfloat: %f\ndouble: %lf\n", u.num, u.fnum, u.dnum);
}

int
main()
{
    union MyUnion u;
    union MyUnion *pu = &u;
    pu->num = 12;
    print_all_members(u);
    pu->fnum = 1.2;
    print_all_members(u);
    pu->dnum = 0.12;
    print_all_members(u);

    return 0;
}
```

Результат выполненной работы:



Задание 2.2

Постановка задачи:

Напишите программу, которая использует union для побайтовой распечатки типа unsigned long.

Математическая модель:

-

Список идентификаторов:

Имя	**Тип**	**Назначение**
u	MyUnion	Некоторое объединение.
i	size_t	Аргумент цилка.

Код программы:

```c
#include <stdio.h>

union MyUnion{
    unsigned long uli;
    char c;
};

int
main()
{
    union MyUnion u;
    u.uli = 1234567890;
    for(size_t i = 0; i < sizeof(unsigned long); i++)
    {
        putchar(u.c);
    }
    putchar('\n');
    return 0;
}
```

Результат работы программы:

Задание 2.3

Постановка задачи:

Создайте перечислимый тип данных (enum) для семи дней недели и распечатайте на экране его значения, как целые числа.

Математическая модель:

-

Список идентификаторов:

Имя	**Тип**	**Назначение**
i	enum Days	Аргумент цикла для вывода дней недели.

Код программы:

```c
#include <stdio.h>

enum Days {
    SUNDAY,
    MONDAY,
    TUESDAY,
    WEDNESDAY,
    THURSDAY,
    FRIDAY,
    SATURDAY
};


int
main()
{
    for(enum Days i = 0; i < 7; i++)
        printf("Day number: %d\n",i );

    return 0;
}
```

Результат работы программы:

```
Day number: 0
Day number: 1
Day number: 2
Day number: 3
Day number: 4
Day number: 5
Day number: 6
```

Задание 2.4

Постановка задачи:

Создайте так называемое размеченное объединение union, которое заключено в виде поля структуры struct вместе с ещё одним полем, которое является перечислением enum и служит индикатором того, что именно на текущий момент хранится в таком вложенном объединении. Создать и заполнить динамический массив таких структур с объединениями внутри, заполняя вспомогательное поле перечисления enum для сохранения информации о хранимом в каждом размеченном объединении типе данных. Реализовать распечатку данных массива таких структур в консоль.

Математическая модель:

-

Список идентификаторов:

Имя	**Тип**	**Назначение**
array	struct MyStruct *	Массив размеченных объединений.
len	int (macros)	Длинна массива array.
iter	struct MyStruct *	Указатель на элемент массива array, используется как аргумент цилка.
n	int	Аргумент цикла для заполнения массива array.
i	int	Аргумент цикла.

Код программы:

```c
#include <stdio.h>
#include <stdlib.h>

#define LEN 10

typedef enum {
    INT_TYPE,
    FLOAT_TYPE,
    POINTER_TYPE
} DataType;

typedef union {
    int i;
    float f;
    void *p;
} Data;

struct MyStruct{
    Data data;
    DataType type;
};

void
printData(struct MyStruct *array, int size) {
    for (int i = 0; i < size; i++) {
        switch (array[i].type) {
            case INT_TYPE:
                printf("%d)int: %d\n", i, array[i].data.i);
                break;
            case FLOAT_TYPE:
                printf("%d)float: %f\n", i, array[i].data.f);
                break;
            case POINTER_TYPE:
                 printf("%d)ptr: %p\n", i, array[i].data.p);
                break;
        }
    }
}

```

```c
int
main()
{
    struct MyStruct *array = NULL;
    array = (struct MyStruct* )malloc(LEN * sizeof(struct MyStruct));
    int n = 12;
    for(struct MyStruct *iter = array; iter < array+LEN; iter++, n++)
    {
        if (n % 3 == INT_TYPE){
            iter->type = INT_TYPE;
            iter->data.i = n;
        }else if (n % 3 == FLOAT_TYPE){
            iter->type = FLOAT_TYPE;
            iter->data.f = n/3.0;
        }else if (n % 3 == POINTER_TYPE){
            iter->type = POINTER_TYPE;
            iter->data.p = iter;
        }
    }
    printData(array, LEN);
    return 0;
}
```

Результат работы программы:

```
0)int: 12
1)float: 4.333333
2)ptr: 0x55878f9562c0
3)int: 15
4)float: 5.333333
5)ptr: 0x55878f9562f0
6)int: 18
7)float: 6.333333
8)ptr: 0x55878f956320
9)int: 21
```