Лабораторная работа №8

1) Тема: Структуры и объединения в языке С (комплект 9 Структуры в С)

9.1)

2) Создайте структуру, которая описывает объект «книга» и содержит в себе 15 основных элементов «дублинского ядра».

Подберите нужные типы данных для этих элементов. Введите эти элементы в экземпляр такой структуры и распечатайте на экране.

3)

```
struct BOOK
{
char title[100];
char creator[100];
char subject[100];
char description[100];
char publisher[100];
char contributor[100];
int date1;
int date2;
int date3;
char type[100];
char format[100];
int identifier;
char source[100];
char language[100];
char coverage[100];
char copyright[100];
}
```

4)

Имя	Смысл	Тип
BOOK	Структура «Книга»	struct
title	Title — название	Массив Элементов типа char
creator	Creator — создатель	Массив Элементов типа char
subject	Subject — тема	Массив Элементов типа char
description	Description — описание	Массив Элементов типа char
publisher	Publisher — издатель	Массив Элементов типа char
contributor	Contributor — внёсший вклад	Массив Элементов типа char
date1	Часть даты	int
date2	Часть даты	int
date3	Часть даты	int
type	Type — тип	Массив Элементов типа char
format	Format — формат документа	Массив Элементов типа char
identifier	Identifier — идентификатор	int
source	Source — источник	Массив Элементов типа char
language	Language — язык	Массив Элементов типа char
coverage	Coverage — покрытие	Массив Элементов типа char
copyright	Rights — авторские права	Массив Элементов типа char
show	Функция, выводящая содержимое объекта структуры	void show (struct BOOK obj)
obj	Объект структуры в функции	struct BOOK obj
objA	Объект структуры	struct BOOK objA

5)

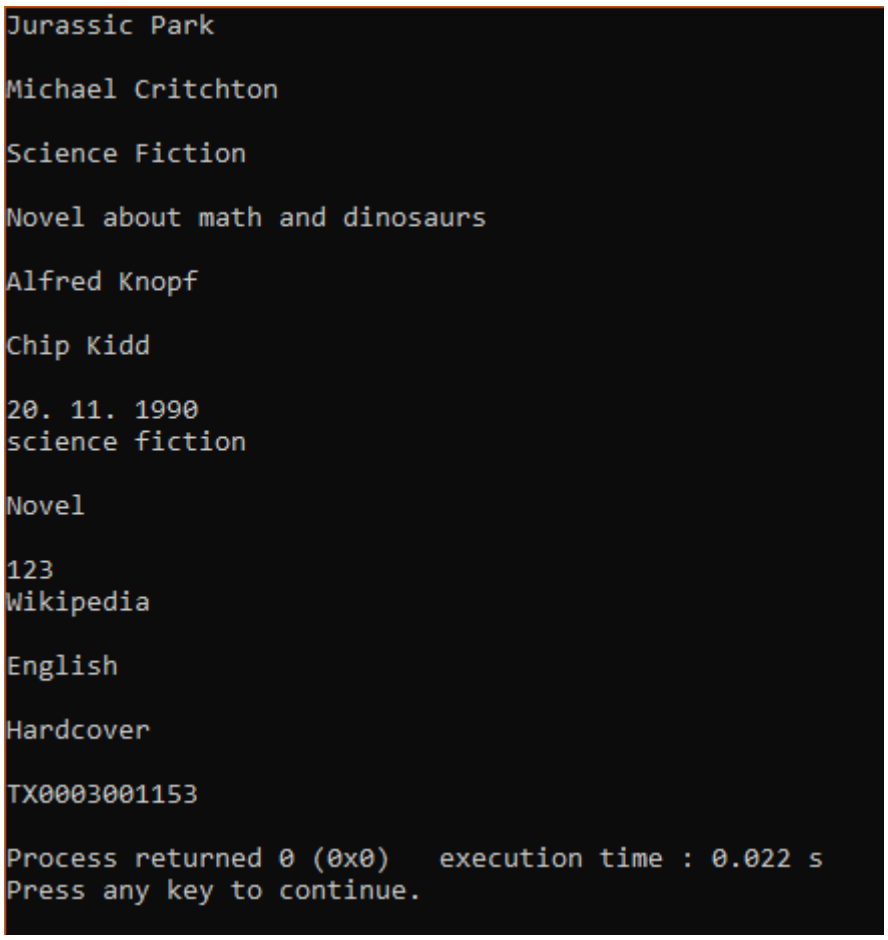
```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
struct BOOK
{
char title[100];
char creator[100];
char subject[100];
char description[100];
char publisher[100];
char contributor[100];
int date1;
int date2;
```

```
int date3;
char type[100];
char format[100];
int identifier;
char source[100];
char language[100];
char coverage[100];
char copyright[100];
};
void show (struct BOOK obj)
{
   puts(obj.title);
   printf("\n");
   puts(obj.creator);
   printf("\n");
   puts(obj.subject);
   printf("\n");
   puts(obj.description);
   printf("\n");
   puts(obj.publisher);
   printf("\n");
   puts(obj.contributor);
   printf("\n");
   printf("%d. %d. %d",obj.date1,obj.date2,obj.date3);
   printf("\n");
   puts(obj.type);
   printf("\n");
   puts(obj.format);
   printf("\n");
   printf("%d",obj.identifier);
```

```
    printf("\n");
    puts(obj.source);
    printf("\n");
    puts(obj.language);
    printf("\n");
    puts(obj.coverage);
    printf("\n");
    puts(obj.copyright);


}
int main()
{
    struct BOOK objA={"Jurassic Park","Michael Critchton","Science Fiction","Novel about math and dinosaurs","Alfred Knopf","Chip Kidd",20,11,1990,"science fiction","Novel",123,"Wikipedia","English","Hardcover","TX0003001153"};
    show(objA);
    return 0;
}
```

6)



9.2)

2)

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

3)

1 вариант

```
struct LOL
{
   int yeet;
   int beet;
   int *weeb;
};
int funky (struct LOL obj)
{
```

```
   int s;
   s=(obj.yeet + obj.beet)*2;
   return s;
}
   int z;
   z=funky(objA);
   objA.weeb=&z
```

2 вариант

```
struct LOL
{
   int yeet;
   int beet;
   int *weeb;
};
int funky (struct LOL obj)
{
   int s;
   s=(obj.yeet + obj.beet)*2;
   obj.weeb=&s;
   return obj.weeb;
}
objA.weeb=funky(objA);
```

4)

Имя	Смысл	Тип
LOL	Структура	struct
yeet	Переменная структуры	int
beet	Переменная структуры	int
*weeb	Указатель структуры	int *
funky	Функция	int funky (struct LOL obj)

obj	Объект структуры в функции	struct LOL obj
s	Результат функции	int
z	Переменная	int
objA	Объект структуры	struct LOL objA

5.1)

```
#include<stdlib.h>
#include<stdio.h>
#include<math.h>
struct LOL
{
   int yeet;
   int beet;
   int *weeb;
};
int funky (struct LOL obj)
{
   int s;
   s=(obj.yeet + obj.beet)*2;
   return s;
}
int main ()
{
   int z;
   struct LOL objA;
   objA.yeet=5;
   objA.beet=10;
   z=funky(objA);
   objA.weeb=&z;
   printf("%d",*objA.weeb);
   return 0;
```

}

6.1)

```
30
Process returned 0 (0x0)   execution time : 0.013 s
Press any key to continue.
```

5.2)

```
#include<stdlib.h>
#include<stdio.h>
#include<math.h>
struct LOL
{
   int yeet;
   int beet;
   int *weeb;
};
int funky (struct LOL obj)
{
   int s;
   s=(obj.yeet + obj.beet)*2;
   obj.weeb=&s;
   return obj.weeb;
}
int main ()
{
   int z;
   struct LOL objA;
   objA.yeet=5;
   objA.beet=10;
   objA.weeb=funky(objA);
   printf("%d",*objA.weeb);
   return 0;
```

}

6.2)

```
30
Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

9.3)

2)

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 bproblems консоли.

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

3)

```
struct CROCO
{
   char name[100];
   int x;
   int y;
   int z;
};
double modul (struct CROCO obj)
{
   double s;
   double q;
   double k;
   q=obj.x*obj.x + obj.y*obj.y + obj.z*obj.z;
   s=sqrt(q);
   return s;
```

```
}
double skal (struct CROCO objA, struct CROCO objB)
{
   double t;
   t=objA.x*objB.x+objA.y*objB.y+objA.z*objB.z;
   return t;
}
double vekt (double a, double b, int c)
{
   double R;
   R=a*b*cosf(c);
   return R;
}
void show(struct CROCO obj)
{
   puts(obj.name);
   printf("%d\n",obj.x);
   printf("%d\n",obj.y);
   printf("%d\n",obj.z);
}
```

4)

Имя	Смысл	Тип
CROCO	Структура обозначающая векторы в 3-х мерном пространстве	struct
name	Название вектора	Массив элементов типа char
x	Координата x структуры	int
y	Координата y структуры	int
z	Координата z структуры	int
modul	Функция нахождения длины вектора	double modul (struct CROCO obj)
obj	Объект структуры функции	struct CROCO obj
s	Результат функции	double
q	Промежуточная переменная функции	double

skal	Функция, находящая скалярное произведение векторов	double skal (struct CROCO objA, struct CROCO objB)
objA	Объект структуры функции	struct CROCO objA
objB	Объект структуры функции	struct CROCO objB
t	Результат функции	double
vekt	Функция, находящая векторное произведение векторов	double vekt (double a, double b, int c)
a	Промежуточная переменная функции	Double
b	Промежуточная переменная функции	Double
c	Промежуточная переменная функции	Int
R	Результат функции	Double
show	Функция, распечатывающая вектор	void show(struct CROCO obj)
*locale	Локаль	char *
corner	Угол между векторами	Int
modA	Длина вектора A	Double
modB	Длина вектора B	Double
skalAB	Скалярное произведение векторов	Double
vektAB	Векторное произведение векторов	Double
objA	Объект A структуры(вектор)	struct CROCO objA
objB	Объект B структуры(вектор)	struct CROCO objB

5)

```
#include<stdlib.h>
#include<stdio.h>
#include<math.h>
#include <locale.h>
struct CROCO
{
   char name[100];
   int x;
   int y;
   int z;
};
double modul (struct CROCO obj)
{
   double s;
   double q;
```

```
    double k;
    q=obj.x*obj.x + obj.y*obj.y + obj.z*obj.z;
    s=sqrt(q);
    return s;
}
double skal (struct CROCO objA, struct CROCO objB)
{
    double t;
    t=objA.x*objB.x+objA.y*objB.y+objA.z*objB.z;
    return t;
}
double vekt (double a, double b, int c)
{
    double R;
    R=a*b*cosf(c);
    return R;
}
void show(struct CROCO obj)
{
    puts(obj.name);
    printf("%d\n",obj.x);
    printf("%d\n",obj.y);
    printf("%d\n",obj.z);
}

int main ()
{
    char *locale=setlocale(LC_ALL, "");
    int corner;
    double modA, modB,skalAB,vektAB;
```

```
    struct CROCO objA={"Vec 1"};
    struct CROCO objB={"Vec 2"};
    printf ("Enter Vector 1\n");
    scanf("%d",&objA.x);
    scanf("%d",&objA.y);
    scanf("%d",&objA.z);
    printf("Enter Vector 2\n");
    scanf("%d",&objB.x);
    scanf("%d",&objB.y);
    scanf("%d",&objB.z);
    printf("Put here a corner between vectors\n");
    scanf ("%d",&corner);
    modA=modul(objA);
    modB=modul(objB);
    printf("Modul A = %f \n",modA);
    printf("Modul B = %f \n",modB);
    skalAB=skal(objA,objB);
    printf("Scalar product = %f\n",skalAB);
    vektAB=vekt(modA,modB,corner);
    printf("Vector product = %f\n\n",vektAB);
    show(objA);
    show(objB);
    return 0;
}
```

6)

```
Enter Vector 1
1
2
3
Enter Vector 2
5
10
12
Put here a corner between vectors
30
Modul A = 3,741657
Modul B = 16,401219
Scalar product = 61,000000
Vector product = 9,466063

Vec 1
1
2
3
Vec 2
5
10
12

Process returned 0 (0x0)   execution time : 10.007 s
Press any key to continue.
```

9.4)

2)

Реализовать в виде структур *двунаправленный связный список* и совершить отдельно его обход в прямом и обратном направлениях с распечаткой значений каждого элемента списка. Использовать структуры и указатели.

3)

```
struct ListElem
{
   int chislo;
   char word[100];
   struct ListElem *prev;
   struct ListElem *next;
};
```

```
struct ListBody
{
   struct ListElem *start;
   struct ListElem *fin;
   unsigned int size;
};

struct ListBody create()
{
   struct ListBody list = {.start=NULL, .fin=NULL, .size = 0};

   return list;
};

void add_to_back(struct ListBody *list, char *word, int l)
{
   struct ListElem *newer = (struct ListElem *) malloc(sizeof(struct ListElem));
   strcpy(newer->word, word);
   newer->chislo = l;
   newer->prev = list->fin;
   newer->next = NULL;
   if (list->fin == NULL)
   {
      list->start = newer;
   }
   else
      {
      list->fin->next = newer;
      }
```

```
    list->fin = newer;
    list->size += 1;
}

void add_to_front(struct ListBody *list, char *word, int l)
{
    struct ListElem *newer = (struct ListElem *) malloc(sizeof(struct ListElem));
    strcpy(newer->word, word);
    newer->chislo = l;
    newer->prev = NULL;
    newer->next = list->start;
    if (list->start == NULL)
    {
        list->fin = newer;
    }
    else
        {
        list->start->prev = newer;
        }
    list->start = newer;
    list->size += 1;
}
```

4)

Имя	Смысл	Тип
ListElem	Элемент двунаправленного списка(структура)	Struct
chislo	Число, хранимое в элементе списка	Int
word	Слово, хранимое в элементе списка	Массив элементов типа char
*prev	Указатель на предыдущий элемент списка	struct ListElem *
*next	Указатель на следующий элемент списка	struct ListElem *
ListBody	Двунаправленный список(структура)	struct
*start	Указатель на начальный элемент списка	struct ListElem *

*fin	Указатель на последний элемент списка	struct ListElem *
size	Размер списка	unsigned int
create	Функция создания списка	struct ListBody create()
list	Результат выполнения функции	struct ListBody
add_to_back	Функция добавления элемента в конец	void add_to_back(struct ListBody *list, char *word, int l)
*list	Список, в который мы добавляем элемент	struct ListBody *
*word	Слово, хранимое в элементе списка (в функции)	char *
l	Число, хранимое в элементе списка (в функции)	Int
*newer	Новый элемент списка (в функции)	struct ListElem *
add_to_front	Функция добавления элемента в начало	void add_to_front(struct ListBody *list, char *word, int l)
boop	Двунаправленный список	struct ListBody boop
*iter	Указатель на список	struct ListElem *
*biter	Указатель на список	struct ListElem *

5)

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

struct ListElem
{
   int chislo;
   char word[100];
   struct ListElem *prev;
   struct ListElem *next;
};

struct ListBody
```

```
{
    struct ListElem *start;
    struct ListElem *fin;
    unsigned int size;
};

struct ListBody create()
{
    struct ListBody list = {.start=NULL, .fin=NULL, .size = 0};

    return list;
};

void add_to_back(struct ListBody *list, char *word, int l)
{
    struct ListElem *newer = (struct ListElem *) malloc(sizeof(struct ListElem));
    strcpy(newer->word, word);
    newer->chislo = l;
    newer->prev = list->fin;
    newer->next = NULL;
    if (list->fin == NULL)
    {
        list->start = newer;
    }
    else
        {
        list->fin->next = newer;
        }
    list->fin = newer;
    list->size += 1;
```

```
}

void add_to_front(struct ListBody *list, char *word, int l)
{
   struct ListElem *newer = (struct ListElem *) malloc(sizeof(struct ListElem));
   strcpy(newer->word, word);
   newer->chislo = l;
   newer->prev = NULL;
   newer->next = list->start;
   if (list->start == NULL)
   {
      list->fin = newer;
   }
   else
      {
      list->start->prev = newer;
      }
   list->start = newer;
   list->size += 1;
}

int main()
{
 struct ListBody boop=create();
 add_to_front(&boop,"Pineapple",2);
 add_to_back(&boop,"Apple",3);
 add_to_back(&boop,"Pen",4);
 add_to_front(&boop,"Pen",1);
 printf("From start to finish \n");
 struct ListElem *iter = boop.start;
```

```
while(iter != NULL)
{
    printf(" - Word number %d is %s \n",iter->chislo,iter->word);
    iter = iter->next;
}
printf ("From finish to start \n");
struct ListElem *biter = boop.fin;
while(biter != NULL)
{
    printf(" - Word number %d is %s \n",biter->chislo,biter->word);
    biter = biter->prev;
}
return 0;
}
```

6)

```
From start to finish
 - Word number 1 is Pen
 - Word number 2 is Pineapple
 - Word number 3 is Apple
 - Word number 4 is Pen
From finish to start
 - Word number 4 is Pen
 - Word number 3 is Apple
 - Word number 2 is Pineapple
 - Word number 1 is Pen

Process returned 0 (0x0)   execution time : 0.017 s
Press any key to continue.
```

9.5)

2)

Используя так называемые «битовые» поля в структуре C, создать экономную структуру в оперативной памяти для заполнения даты некоторого события, например даты рождения человека. Ссылки на описание использования битовых полей:

3)

```
struct S {
 int Day : 6;
 int Month : 4;
 int Year : 12;
};
```

4)

Имя	Смысл	Тип
S	Структура	struct
Day	Битовое поле для дня в структуре	int
Month	Битовое поле для месяца в структуре	int
Year	Битовое поле для года в структуре	int
gregg	Объект в структуре (Дата)	struct S gregg

5)

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

struct S {
 int Day : 6;
 int Month : 4;
 int Year : 12;
};
int main()
{
    struct S gregg;
    gregg.Day=24;
    gregg.Month=6;
    gregg.Year=2001;
    printf("Gregg's B day is %d.0%d.%d",gregg.Day,gregg.Month,gregg.Year);
```

```
    return 0;
}
```

6)

```
Gregg's B day is 24.06.2001
Process returned 0 (0x0)   execution time : 0.014 s
Press any key to continue.
```