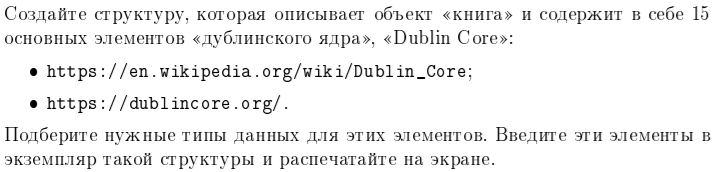
**Лабораторная работа №8**

**Часть 1**

**Задача 1**

Постановка задачи:



Математическая модель: не требуется

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| DublinCore | struct | Структура |
| title | char | Поле структуры |
| creator | char | Поле структуры |
| subject | char | Поле структуры |
| description | char | Поле структуры |
| publisher | char | Поле структуры |
| contributer | char | Поле структуры |
| day | int | Поле структуры |
| month | int | Поле структуры |
| year | int | Поле структуры |
| type | char | Поле структуры |
| format | char | Поле структуры |
| identifier | char | Поле структуры |
| source | char | Поле структуры |
| language | char | Поле структуры |
| relation | char | Поле структуры |
| coverage | char | Поле структуры |
| rights | char | Поле структуры |
| book | struct | Экземпляр структуры |

Код программы:

#include <stdio.h>

#include <string.h>

struct DublinCore {

    char title[30];

    char creator[30];

    char subject[30];

    char description[100];

    char publisher[30];

    char contributer[30];

    int day;

    int month;

    int year;

    char type[10];

    char format[10];

    char identifier[50];

    char source[30];

    char language[10];

    char relation[40];

    char coverage[30];

    char rights[50];

} book;

int main(void) {

    strcpy(book.title, "Essential C");

    strcpy(book.creator, "Nick Parlante");

    strcpy(book.subject, "Programming on C-language");

    strcpy(book.description, "This Stanford CS Education document tries to summarize all the basic features of the C language");

    strcpy(book.publisher, "Stanford CS Education Library");

    strcpy(book.contributer, "Negar Shamma and A. P. Garcia");

    book.day = 15;

    book.month = 4;

    book.year = 2003;

    strcpy(book.type, "Textbook");

    strcpy(book.format, "E-book");

    strcpy(book.identifier, "http://cslibrary.stanford.edu/101/EssentialC.pdf");

    strcpy(book.source, "This is the full original text");

    strcpy(book.language, "English");

    strcpy(book.relation, "https://codernet.ru/books/c\_plus/");

    strcpy(book.coverage, "Actual for nowadays");

    strcpy(book.rights, "Copyright 1996-2003, Nick Parlante");

    printf("Title: %s\n", book.title);

    printf("Creator: %s\n", book.creator);

    printf("Subject: %s\n", book.subject);

    printf("Description: %s\n", book.description);

    printf("Publisher: %s\n", book.publisher);

    printf("Contributer: %s\n", book.contributer);

    printf("Date: %.2d.%.2d.%.4d\n", book.day, book.month, book.year);

    printf("Type: %s\n", book.type);

    printf("Format: %s\n", book.format);

    printf("Identifier: %s\n", book.identifier);

    printf("Source: %s\n", book.source);

    printf("Language: %s\n", book.language);

    printf("Relation: %s\n", book.relation);

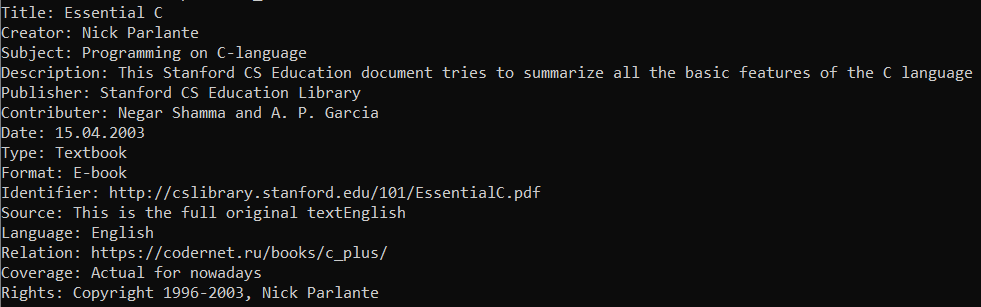
    printf("Coverage: %s\n", book.coverage);

    printf("Rights: %s\n", book.rights);

    return 0;

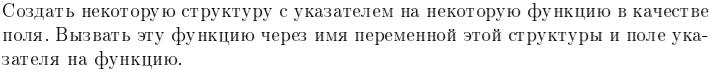
}

Результат:



**Задача 2**

Постановка задачи:



Математическая модель:

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| factorial | int | Функция вычисления факториала |
| result | int | Результат функции |
| n | int | Аргумент функции |
| i | int | Параметр цикла |
| StructFunc | struct | Структура |
| \*f | int | Поле структуры |
| obj | struct | Экземпляр структуры |

Список идентификаторов:

Код:

#include <stdio.h>

#include <stdlib.h>

int factorial(int n) {

    int result = 1;

    for (int i = 1; i <= n; i++)

        result \*= i;

    return result;

}

struct StructFunc {

    int(\*f)(int);

} obj;

int main(void) {

    obj.f = factorial;

    int a;

    printf("Factorial what a number need to calculate: ");

    scanf("%d", &a);

    printf("%d! = %d", a, obj.f(a));

    return 0;

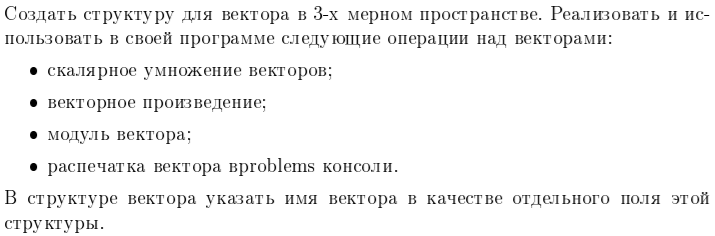
}

Результат:



**Задача 3**

Постановка задачи:



Математическая модель:

* Скалярное произведение векторов А • B = a1b1 + *а2b2 + а3b3* представ­ляет собой сумму попарных произведений соответствующих коор­динат исходных векторов.
* Векторным произведением векторов АхВ = (а2b3 - *b2a3, a3b1 - b3а1, а1b2* - b1a2) является вектор, координаты которого *ck* вычисляются как *ck = ak+1bk+2 —* ak+2bk+1 (k=1,2,3) с учетом циклической переста­новки индексов (то есть *а4=а1, а5=а2,* и так же для компонентов *bk).*
* Модулем вектора является число, равное корню квадратному из скалярного произведения вектора на самого себя: |А| =

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| Vector | struct | Структура |
| name | char | Поле структуры |
| x | int | Поле структуры |
| y | int | Поле структуры |
| z | int | Поле структуры |
| v1 | struct | Экземпляр структуры |
| v2 | struct | Экземпляр структуры |
| scalarProd | int | Функция скалярного умножения векторов |
| moduleVector | double | Функция нахождения модуля вектора |
| vectorProd | struct | Функция нахождения векторного умнож. |
| a | struct | Аргумент функции (Экземпляр структуры) |
| b | struct | Аргумент функции (Экземпляр структуры) |
| c | struct | Аргумент функции (Экземпляр структуры) |

Код:

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <string.h>

struct Vector {

    char name[1];

    int x;

    int y;

    int z;

} v1, v2;

int scalarProd(struct Vector a, struct Vector b) {

    return a.x \* b.x + a.y \* b.y + a.z \* b.z;

}

double moduleVector(struct Vector a) {

    return sqrt(a.x \* a.x + a.y \* a.y + a.z \* a.z);

}

struct Vector vectorProd(struct Vector a, struct Vector b) {

    struct Vector c;

    c.x = a.y \* b.z - a.z \* b.y;

    c.y = a.z \* b.x - a.x \* b.z;

    c.z = a.x \* b.y - a.y \* b.x;

    return c;

}

int main(void) {

    printf("Enter the name and coordinates of 1st vector: ");

    scanf("%s %d %d %d", &v1.name, &v1.x, &v1.y, &v1.z);

    printf("Enter the name and coordinates of 2nd vector: ");

    scanf("%s %d %d %d", &v2.name, &v2.x, &v2.y, &v2.z);

    struct Vector result;

    printf("Scalar production %s \* %s = %d\n", v1.name, v2.name, scalarProd(v1, v2));

    result = vectorProd(v1, v2);

    printf("Vector production %s \* %s: = {%d, %d, %d}\n", v1.name, v2.name, result.x, result.y, result.z);

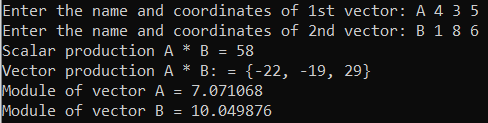
    printf("Module of vector %s = %lf\n", v1.name, moduleVector(v1));

    printf("Module of vector %s = %lf\n", v2.name, moduleVector(v2));

    return 0;

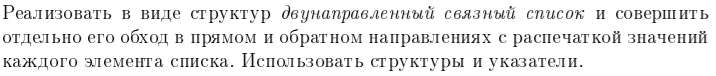
}

Результат:



**Задача 4**

Постановка задачи:



Математическая модель: ---

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| ListNode | struct | Структура |
| num | int | Номер элемента |
| \*previous | struct | Экземпляр структуры как поле (указатель на след элемент) |
| \*next | struct | Экземпляр структуры как поле (указатель на предыдущий элемент) |
| LinkedList | struct | Структура |
| \*head | struct | Экземпляр структуры как поле |
| \*tail | struct | Экземпляр структуры как поле |
| size | unsigned int | Размер списка |
| create\_list | struct | Функция создания пустого списка |
| push\_to\_end | void | Функция добавления элемента в конец |
| \*new\_node | struct | Новый элемент |
| my\_list | struct | Связный список |
| \*iterator | struct | Итератор цикла |
| n | const | Количество элементов в списке |

Код:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define n 10

typedef unsigned int uint;

struct ListNode {

    int num;

    struct ListNode \*previous;

    struct ListNode \*next;

};

struct LinkedList {

    struct ListNode \*head;

    struct ListNode \*tail;

    uint size;

};

struct LinkedList create\_list() {

    struct LinkedList list = {

        .head = NULL,

        .tail = NULL,

        .size = 0

    };

    return list;

}

void push\_to\_end(struct LinkedList \*list, int num) {

    struct ListNode \*new\_node = (struct ListNode \*)malloc(sizeof(struct ListNode));

    new\_node->num = num;

    new\_node->previous = list->tail;

    new\_node->next = NULL;

    if (list-> tail == NULL)

        list->head = new\_node;

    else

        list->tail->next=new\_node;

    list->tail = new\_node;

    list->size += 1;

}

int main(void) {

    struct LinkedList MyList = create\_list();

    for (int i = 1; i <= n; i++) {

        push\_to\_end(&MyList, i);

    }

    struct ListNode \*iterator = MyList.head;

    while(iterator != NULL) {

        printf("Element %d: previous = %p; this = %p; next = %p\n", iterator->num, iterator->previous, iterator, iterator->next);

        iterator = iterator->next;

    }

    printf("\n\n");

    iterator = MyList.tail;

    while(iterator != NULL) {

        printf("Element %d: previous = %p; this = %p; next = %p\n", iterator->num, iterator->previous, iterator, iterator->next);

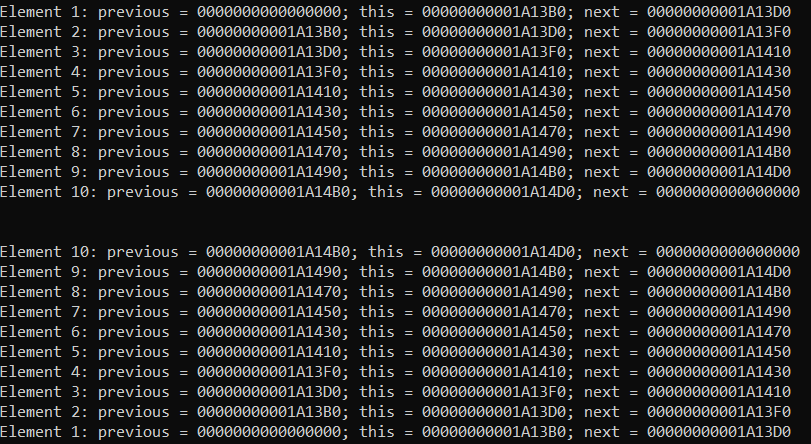
        iterator = iterator->previous;

    }

    return 0;

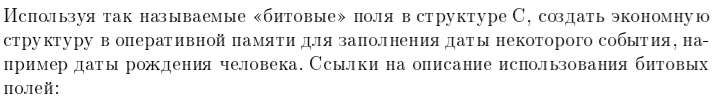
}

Результат:



**Задача 5**

Постановка задачи:



Математическая модель:

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| date | struct | Структура |
| day | unsigned int | Поле структуры / день |
| month | unsigned int | Поле структуры / месяц |
| year | unsigned int | Поле структуры / год |
| obj1 | struct | Экземпляр структуры |

Код:

#include <stdio.h>

#include <stdlib.h>

struct date {

    unsigned int day: 5;

    unsigned int month: 4;

    unsigned int year: 11;

};

int main(void) {

    unsigned int day, month, year;

    printf("Enter day: ");

    scanf("%d", &day);

    printf("Enter month: ");

    scanf("%d", &month);

    printf("Enter year: ");

    scanf("%d", &year);

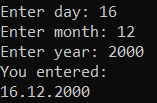
    struct date obj1 = {day, month, year};

    printf("You entered:\n%.2d.%.2d.%d", obj1.day, obj1.month, obj1.year);

    return 0;

}

Результат:



**Задача 6**

Постановка задачи:

Математическая модель:

Список идентификаторов:

Код:

Результат:

**Часть 2**

**Задача 1**

Постановка задачи:



Математическая модель:

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| MyUnion | union | Объединение |
| num | int | Поле объединения |
| symbol | char | Поле объединения |
| obj | union | Экземпляр объединения |
| point | union | Указатель на объединение |
| i | int | Параметр цикла |

Код:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

union MyUnion {

  int num;

  char symbol;

};

int main(void) {

  union MyUnion obj = {0};

  union MyUnion \*point = &obj;

  for (int i = 0; i <= 16; i++) {

    point->num = i;

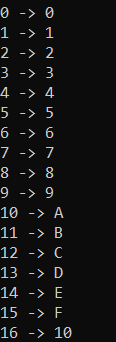
    printf("%d -> %X\n", obj.num, obj.symbol);

  }

  return 0;

}

Результат:



**Задача 2**

Постановка задачи:



Математическая модель:

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| byteUnion | union | Объединение |
| num | unsigned long | Поле объединения |
| str | unsigned char | Поле объединения |
| byte | union | Экземпляр объединения |
| i | int | Параметр цикла |

Код:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef union byteUnion

{

    unsigned long num;

    unsigned char str[3];

}obj;

int main(void) {

    obj byte = { 4294967295 };

    for (int i = 0; i < 4; i++)

        printf("%ld\t", (int)byte.str[i]);

    return 0;

}

Результат:



**Задача 3**

Постановка задачи:



Математическая модель: ---

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| weekDays | enum | Перечисляемый тип данных, содержащий дни недели |
| Sunday | int | Элемент weekDays |
| Monday | int | Элемент weekDays |
| Tuesday | int | Элемент weekDays |
| Wednesday | int | Элемент weekDays |
| Thursday | int | Элемент weekDays |
| Friday | int | Элемент weekDays |
| Saturday | int | Элемент weekDays |
| day0 | int | День 0 |
| day1 | int | День 1 |
| day2 | int | День 2 |
| day3 | int | День 3 |
| day4 | int | День 4 |
| day5 | int | День 5 |
| day6 | int | День 6 |

Код:

#include <stdio.h>

#include <stdlib.h>

enum weekDays {

    Sunday,

    Monday,

    Tuesday,

    Wednesday,

    Thursday,

    Friday,

    Saturday

};

int main(void) {

    enum weekDays day0, day1, day2, day3, day4, day5, day6;

    day0 = Sunday; day1 = Monday; day2 = Tuesday;

    day3 = Wednesday; day4 = Thursday; day5 = Friday; day6 = Saturday;

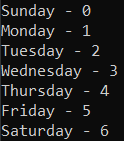
    printf("Sunday - %d\nMonday - %d\nTuesday - %d\nWednesday - %d\n"

    "Thursday - %d\nFriday - %d\nSaturday - %d\n", day0, day1, day2, day3, day4, day5, day6);

    return 0;

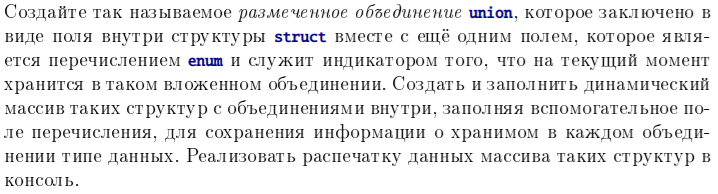
}

Результат:



**Задача 4**

Постановка задачи:



Математическая модель:

Список идентификаторов:

Код:

Результат: