Министерство образования и науки Российской федерации

Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Санкт-Петербургский Государственный Электротехнический Университет «ЛЭТИ» им. В. И. Ульянова (Ленина)» (СПбГЭТУ «ЛЭТИ»)

Факультет компьютерных технологий и информатики

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Кафедра вычислительной техники**

**Отчёт по лабораторной работе №1**

по дисциплине «Программирование»

на тему

«Структуры С/C++»

Выполнил студент гр.5307 Самоуков Н.В.

Проверил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"Выполнено" "\_\_\_\_"\_\_\_\_\_\_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Санкт-Петербург**

**2015 г.**

Оглавление

[Цель работы 3](#_Toc448654107)

[1.Задание 3](#_Toc448654108)

[2.Уточнение задания 3](#_Toc448654109)

[3.Контрольные примеры 4](#_Toc448654110)

[4.Описание структур 4](#_Toc448654111)

[4.1 Описание полей структуры SHIP 4](#_Toc448654112)

[5.Описание переменных главной функции 5](#_Toc448654113)

[6.Краткое описание алгоритма 5](#_Toc448654114)

[7.Иерархическая структура программы 6](#_Toc448654115)

[8.Схема алгоритма главной функции 7](#_Toc448654116)

[9.Описание функций 11](#_Toc448654117)

[9.1 Функция my\_getch 11](#_Toc448654118)

[9.2 Функция input\_number 11](#_Toc448654119)

[9.3 Функция input\_yn 12](#_Toc448654120)

[9.4 Функция input\_string 13](#_Toc448654121)

[9.5 Функция str\_len 13](#_Toc448654122)

[9.6 Функция free\_structs 14](#_Toc448654123)

[9.7 Функция print\_struct 15](#_Toc448654124)

[9.8 Функция print\_structs 15](#_Toc448654125)

[9.9 Функция get\_type 16](#_Toc448654126)

[9.10 Функция get\_name 17](#_Toc448654127)

[9.11 Функция get\_coord 18](#_Toc448654128)

[9.12 Функция get\_struct 18](#_Toc448654129)

[9.13 Функция get\_structs 19](#_Toc448654130)

[9.14 Функция processing 20](#_Toc448654131)

[10.Текст программы 21](#_Toc448654132)

[11.Результат 29](#_Toc448654133)

[Вывод 29](#_Toc448654134)

# Цель работы

Получить практические навыки использования структур в языке С/С++.

# 

# 1.Задание

Разработать алгоритм и написать программу, выполняющую многократно по желанию пользователя ввод массива структур, обработку массива структур и вывод результата.

# 

# 2.Уточнение задания

Обработанный массив состоит из элементов исходного массива удовлетворяющих задаваемым критериям.

# 

# 3.Контрольные примеры

Контрольные примеры представлены в таблице 1. Контрольные примеры.

Таблица 1. Контрольные примеры

|  |  |  |
| --- | --- | --- |
| № | Входные данные | Ожидаемый результат |
| 1 | ship[0]={"Ship1","boat",7,7,7}  ship[1]={"Ship2","steelship",4,-5,10}  ship[2]={"Ship3","boat",6,60,5}  name\_find="Ship1" | ship\_result[0]={"Ship1","boat",7,7,7} |
| 2 | ship[0]={"Ship1","boat",7,7,7}name\_find="Ship2" | ship\_result=NULL |
| 3 | ship[0]={"Ship1","boat",7,7,7}  ship[1]={"Ship2","steelship",4,-5,10}  ship[2]={"Ship3","boat",6,60,5}  type\_find="boat" | ship\_result [0]={"Ship1","boat",7,7,7}  ship\_result [1]={"Ship3","boat",6,60,5} |

# 4.Описание структур

# 4.1 Описание полей структуры SHIP

Описание полей структуры SHIP ,приведено в таблице 2 Описание полей структуры SHIP.

Таблица 2. Описание полей структуры SHIP.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| char \* | name | Указатель на первый символ строки, содержащий имя кораблика |
| char \* | type | Указатель на первый символ строки, содержащий тип кораблика |
| int | x | Положение кораблика по x |
| int | y | Положение кораблика по y |
| int | z | Положение кораблика по z |

# 5.Описание переменных главной функции

Переменные, используемые в главной функции, приведены в таблице 3 Переменные, используемые в главной функции.

Таблица 3. Переменные, используемые в главной функции

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| ship | SHIP \* | Указатель на первый элемент исходного массива |
| ship\_result | SHIP \* | Указатель на первый элемент обработанного массива |
| ship\_len | int | Количество элементов в исходном массиве |
| ship\_result\_len | int | Количество элементов в обработанном массиве |
| menu | int | Выбранный пункт меню |
| is\_input | bool | Флажок наличия исходного массива (1-исходный массив введён, 0-исходный массив не введён) |
| is\_processed | bool | Флажок проведения обработки массива (1-обработка была, 0-обработки не было) |
| is\_output | bool | Флажок наличия исходного массива (1-исходный массив введён, 0-исходный массив не введён) |

# 6.Краткое описание алгоритма

Шаг 1. is\_input = false; is\_processed = false; is\_output = false;

Шаг 2. Очистить экран.

Шаг 3. Вывести меню.

Шаг 4. Ввести menu

Шаг 5. Очистить экран.

Шаг 6. Если menu==1 ,то

is\_input = true;

is\_output = false;

is\_processed = false;

Ввести ship.

Шаг 13

Шаг 7. Если menu==2 ,то

Вывести ship или сообщение об отсутствии входных данных.

Шаг 13

Шаг 8. Если menu==3 ,то

Обработать ship и если ship\_result\_len > 0, то is\_output = true;

и вывести сообщение об успешной обработке, иначе

is\_output = false; и сообщение об отсутствии результата.

Шаг 13

Шаг 9. Если menu==4 ,то

ship или соответствующее сообщение об отсутствии входных данных.

Шаг 13

Шаг 10. Если menu==5 ,то

Вывести справку.

Шаг 13

Шаг 11. Если menu==0 ,то

Шаг 13

Шаг 12. Вывести сообщение об ошибке.

Шаг 13. Если menu != 0 то Шаг 2, иначе конец программы.

# 7.Иерархическая структура программы

Иерархическая структура программы представлена на рисунке 1.

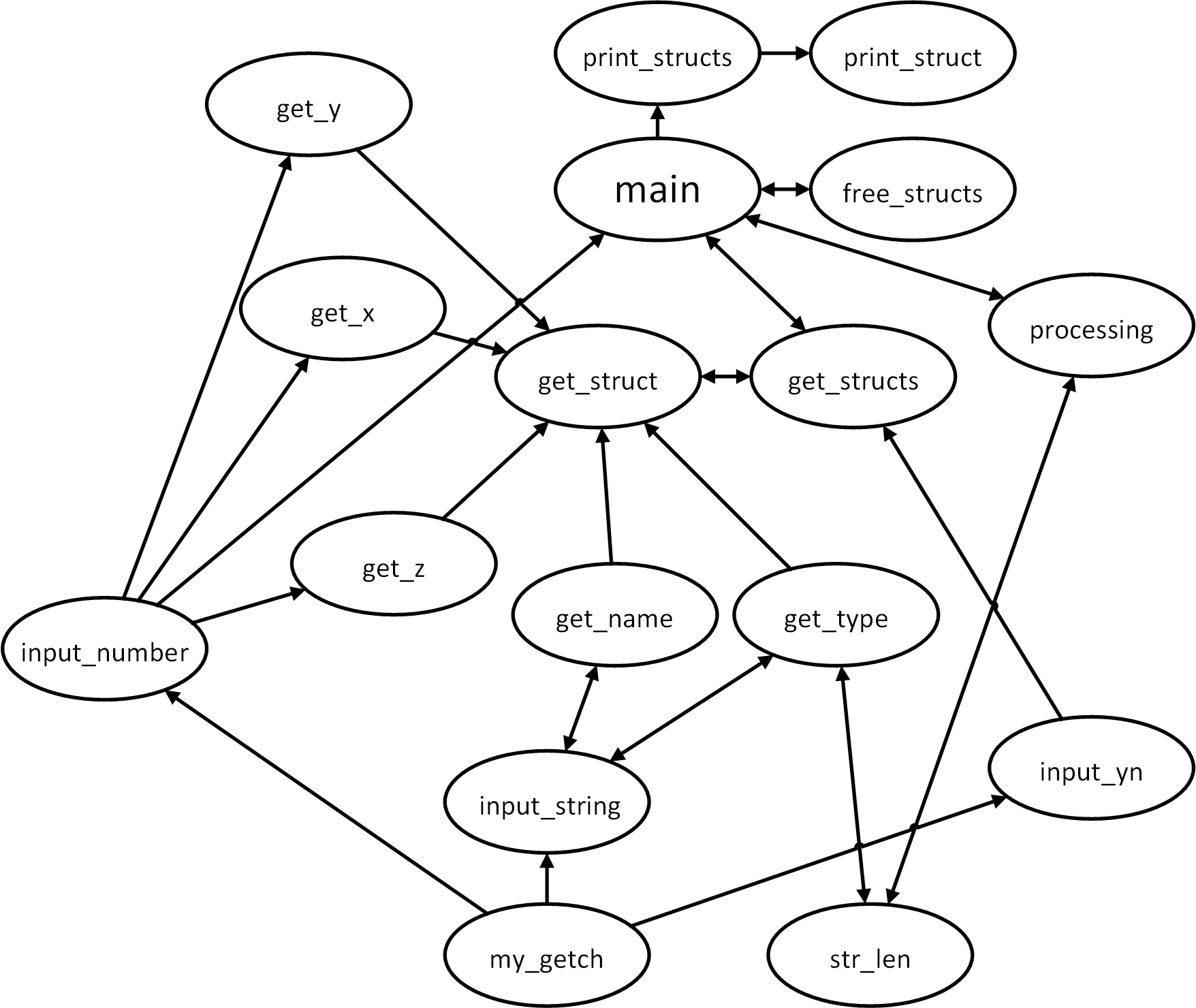


Рисунок 1. Иерархическая структура программы

# 8.Схема алгоритма главной функции

Схема алгоритма представлена на рисунках 2.1-2.4.

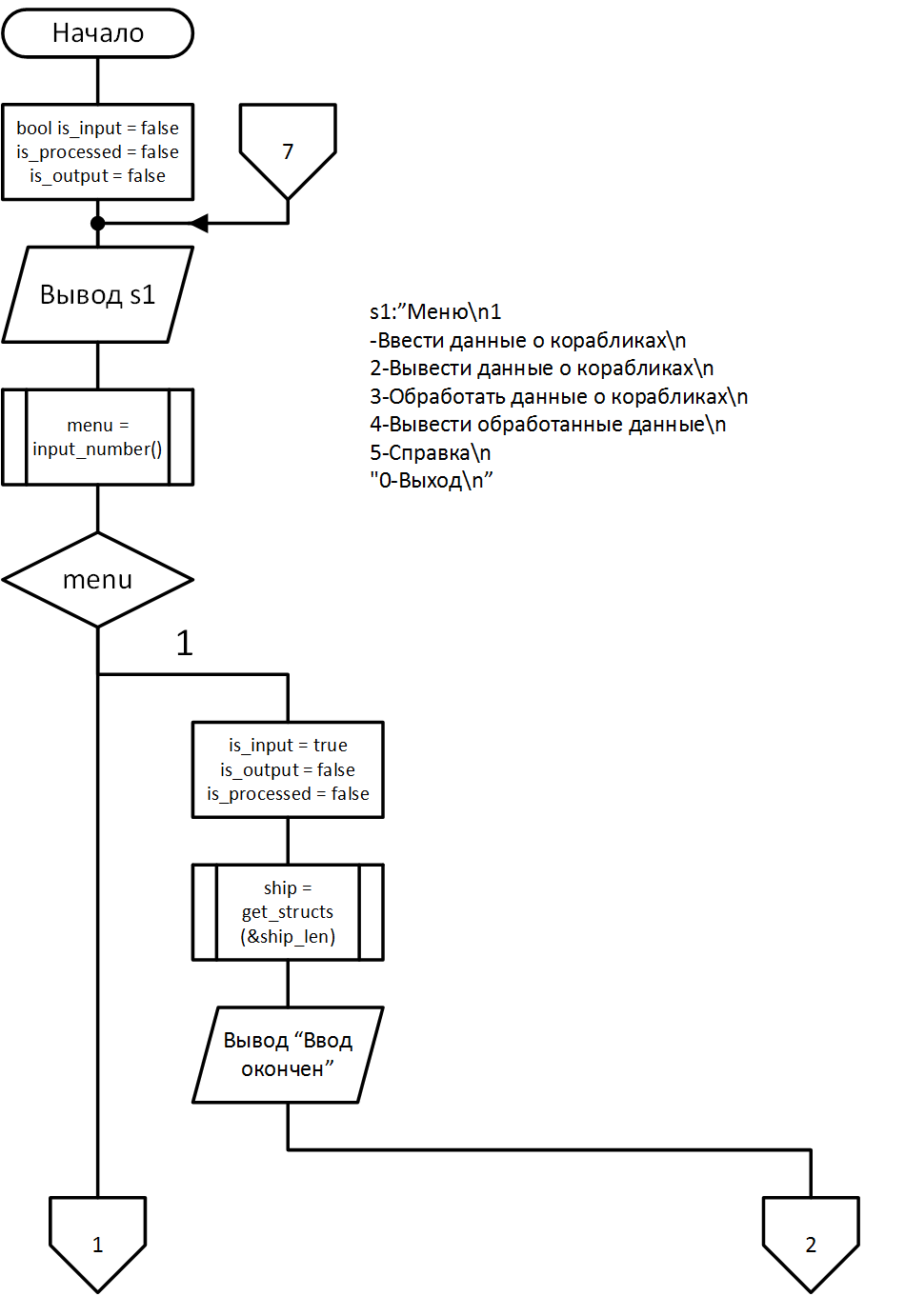


Рисунок 2.1. Схема алгоритма главной функции

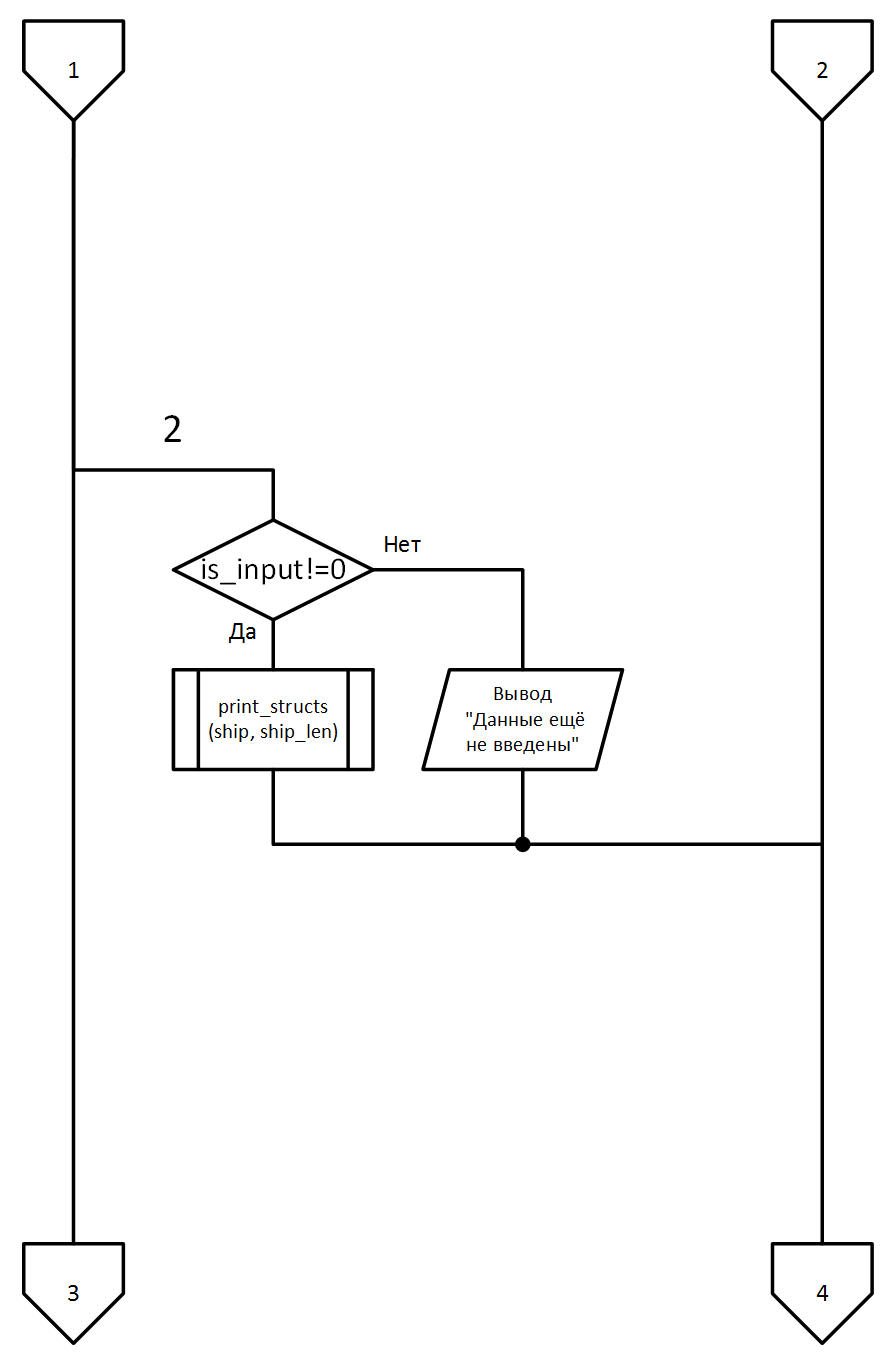


Рисунок 2.2. Схема алгоритма главной функции

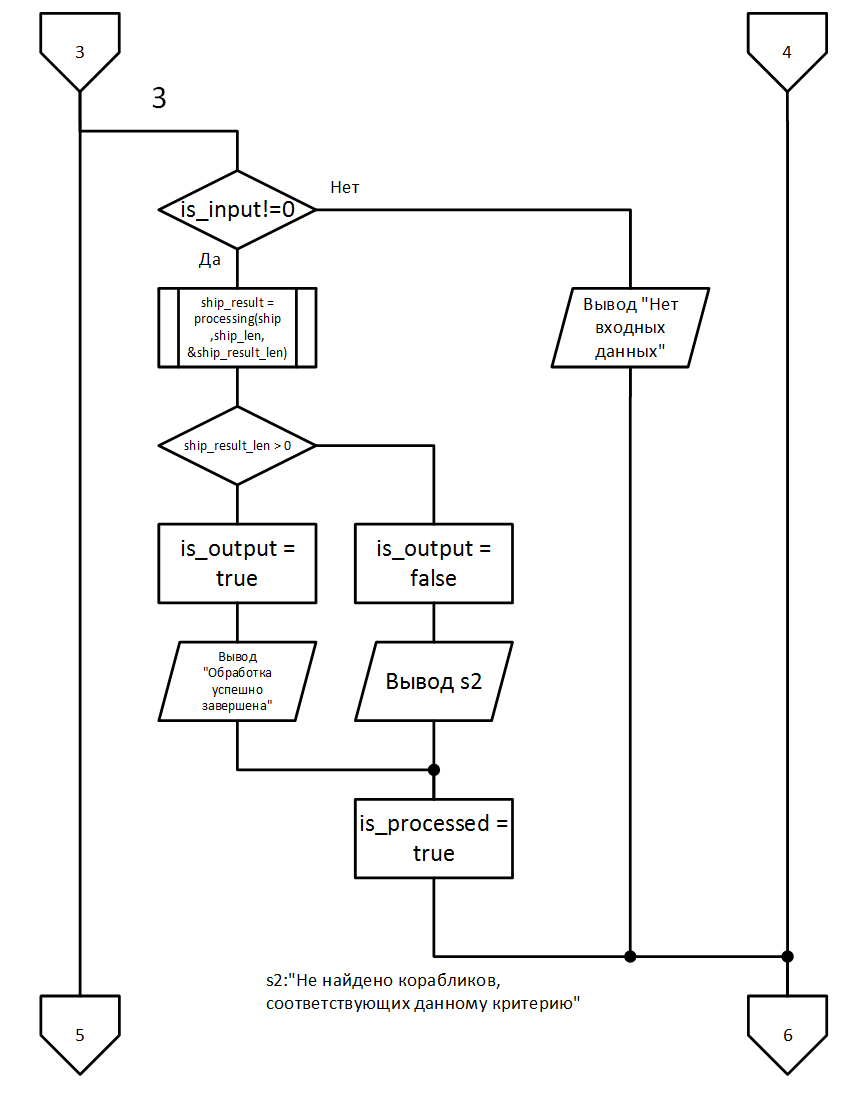


Рисунок 2.3. Схема алгоритма главной функции

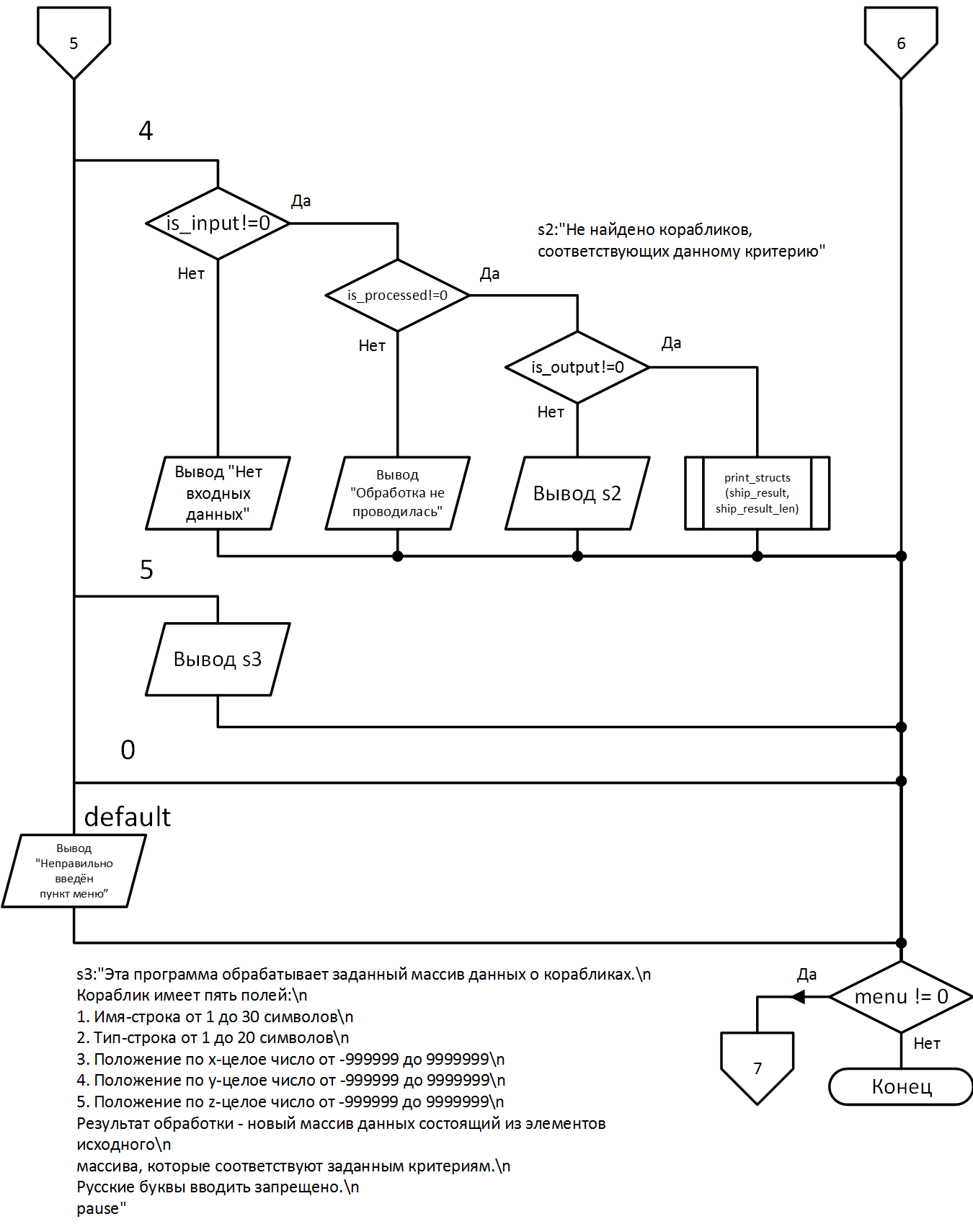


Рисунок 2.4. Схема алгоритма главной функции

# 9.Описание функций

# 9.1 Функция my\_getch

**Назначение функции:**

Это функция для получения нажатой клавиши.

**Прототип:**

char my\_getch();

**Возвращаемое значение:**

Эта функция возвращает нажатую пользователем кнопку.

**Пример вызова:**

ch=my\_getch();

**Описание переменных:**

Описание переменных в таблице 4. Переменные функции my\_getch.

Таблица 4. Переменные функции my\_getch

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | ch | char | Символ нажатой клавиши |
| **Формальные параметры** | Формальные переменные отсутствуют | | |
| **Локальные переменные** | ch | char | Символ нажатой клавиши |

# 9.2 Функция input\_number

**Назначение функции:**

Это функция для ввода целого числа.

**Прототип:**

int input\_number();

**Возвращаемое значение:**

Эта функция возвращает введённое число.

**Пример вызова:**

i=input\_number()

**Описание переменных:**

Описание переменных в таблице 5. Переменные функции str\_emepty

Таблица 5. Переменные функции str\_emepty

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | i | int | Введённое число |
| **Формальные параметры** | Формальные параметры отсутствуют | | |
| **Локальные переменные** | number | int | Введённое число |
| s | char \* | Указатель на первый символ исходной строки |
| length | int | Длина строки |
| ch | char | Введённый символ |

# 9.3 Функция input\_yn

**Назначение функции:**

Это функция для ввода ответа: да или нет.

**Прототип:**

bool input\_yn();

**Возвращаемое значение:**

Эта функция возвращает 0 или 1.

**Пример вызова:**

input\_yn()

**Описание переменных:**

Описание переменных в таблице 6. Переменные функции input\_yn

Таблица 6. Переменные функции input\_yn

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | i | int | Введённое число |
| **Формальные параметры** | Формальные переменные отсутствуют | | |
| **Локальные переменные** | sy | char | Введённый символ |
| ch | char | Нажатая клавиша |

# 9.4 Функция input\_string

**Назначение функции:**

Это функция для ввода строки.

**Прототип:**

char \*input\_string(int size);

**Возвращаемое значение:**

Эта функция возвращает указатель на первый символ введённой строки.

**Пример вызова:**

s=input\_string(30);

**Описание переменных:**

Описание переменных в таблице 7. Переменные функции input\_string

Таблица 7. Переменные функции input\_string

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | size | int | Максимальная длинна строки |
| **Формальные параметры** | s | char \* | Указатель на первый символ введённой строки |
| **Локальные переменные** | s | char \* | Указатель на первый символ исходной строки |
| length | int | Длина строки |
| ch | char | Введённый символ |

# 9.5 Функция str\_len

**Назначение функции:**

Это функция для ввода строки.

**Прототип:**

char \*str\_len (int size);

**Возвращаемое значение:**

Эта функция возвращает длину строки.

**Пример вызова:**

str\_len ("sss")

**Описание переменных:**

Описание переменных в таблице 8. Переменные функции str\_len

Таблица 8. Переменные функции str\_len

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | Фактические переменные отсутствуют | | |
| **Формальные параметры** | Формальные переменные отсутствуют | | |
| **Локальные переменные** | i | int | Длина строки |

# 9.6 Функция free\_structs

**Назначение функции:**

Это функция для освобождения памяти.

**Прототип:**

SHIP \*free\_structs(SHIP \*ship, int len);

**Возвращаемое значение:**

Эта функция возвращает NULL.

**Пример вызова:**

ship=free\_structs(ship,len);

**Описание переменных:**

Описание переменных в таблице 9. Переменные функции free\_structs

Таблица 9. Переменные функции free\_structs

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | ship | SHIP\* | Указатель на первый элемент массива корабликов |
| len | int | Количество элементов в массиве |
| **Формальные параметры** | ship | SHIP\* | Указатель на первый элемент массива корабликов |
| len | int | Количество элементов в массиве |
| **Локальные переменные** | i | int | Вспомогательная переменная |

# 9.7 Функция print\_struct

**Назначение функции:**

Это функция для вывода полей одной структуры.

**Прототип:**

void print\_struct(SHIP ship);

**Возвращаемое значение:**

Эта функция ничего не возвращает .

**Пример вызова:**

print\_struct(ship);

**Описание переменных:**

Описание переменных в таблице 10. Переменные функции print\_struct

Таблица 10. Переменные функции print\_struct

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | ship | SHIP\* | Указатель на первый элемент массива корабликов |
| **Формальные параметры** | ship | SHIP | Структура, содержащая данные о кораблике |
| **Локальные переменные** | Локальные переменные отсутствуют | | |

# 9.8 Функция print\_structs

**Назначение функции:**

Это функция для вывода массива данных о кораблике.

**Прототип:**

void print\_structs(SHIP \*ship,int ships);

**Возвращаемое значение:**

Эта функция ничего не возвращает .

**Пример вызова:**

print\_structs(ship,len);

**Описание переменных:**

Описание переменных в таблице 11. Переменные функции print\_structs

Таблица 11. Переменные функции print\_structs

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | ship | SHIP\* | Указатель на первый элемент массива корабликов |
| len | int | Количество элементов в массиве |
| **Формальные параметры** | ship | SHIP\* | Указатель на первый элемент массива корабликов |
| ships | int | Количество элементов в массиве |
| **Локальные переменные** | i | int | Вспомогательная переменная |

# 9.9 Функция get\_type

**Назначение функции:**

Это функция для ввода типа кораблика.

**Прототип:**

char\* get\_type();

**Возвращаемое значение:**

Эта функция возвращает указатель на первый элемент введённой строки.

**Пример вызова:**

s=get\_type();

**Описание переменных:**

Описание переменных в таблице 12. Переменные функции get\_type

Таблица 12. Переменные функции get\_type

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | s | char \* | Указатель на первый символ строки |
| **Формальные параметры** | Формальные параметры отсутствуют | | |
| **Локальные переменные** | i | int | Вспомогательная переменная |
| s | char\* | Указатель на первый символ вводимой строки |
| type | char | Тип кораблика |
| len | char | Длина строки |
| types | char\* | Массив указателей на первый символ строки |

# 9.10 Функция get\_name

**Назначение функции:**

Это функция для ввода имени кораблика.

**Прототип:**

char\* get\_name();

**Возвращаемое значение:**

Эта функция возвращает указатель на первый элемент введённой строки.

**Пример вызова:**

s= get\_name();

**Описание переменных:**

Описание переменных в таблице 13. Переменные функции get\_name

Таблица 13. Переменные функции get\_name

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | s | char \* | Указатель на первый символ строки |
| **Формальные параметры** | Формальные переменные отсутствуют | | |
| **Локальные переменные** | s | char\* | Указатель на первый символ вводимой строки |
| exit | bool | Флажок выхода |

# 9.11 Функция get\_coord

**Назначение функции:**

Это функция для ввода координат.

**Прототип:**

int get\_coord(char ch);

**Возвращаемое значение:**

Эта функция возвращает введённое число.

**Пример вызова:**

i=get\_coord('z');

**Описание переменных:**

Описание переменных в таблице 14. Переменные функции get\_coord

Таблица 14. Переменные функции get\_coord

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | i | int | Введённое число |
| **Формальные параметры** | ch | char | Название координаты |
| **Локальные переменные** | Локальные переменные отсутствуют | | |

# 9.12 Функция get\_struct

**Назначение функции:**

Это функция для .

**Прототип:**

void get\_struct(SHIP \*info);

**Возвращаемое значение:**

Эта функция ничего не возвращает .

**Пример вызова:**

get\_struct(ship);

**Описание переменных:**

Описание переменных в таблице 15. Переменные функции get\_struct

Таблица 15. Переменные функции get\_struct

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | ship | SHIP\* | Указатель на элемент массива корабликов |
| **Формальные параметры** | info | SHIP\* | Указатель на элемент массива корабликов |
| **Локальные переменные** | Локальные переменные отсутствуют | | |

# 9.13 Функция get\_structs

**Назначение функции:**

Это функция для ввода массива структур.

**Прототип:**

SHIP \*get\_structs(int \*ships);

**Возвращаемое значение:**

Эта функция возвращает указатель на первый элемент массива корабликов .

**Пример вызова:**

ship=get\_structs(&len);

**Описание переменных:**

Описание переменных в таблице 16. Переменные функции get\_structs

Таблица 16. Переменные функции get\_structs

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | ship | SHIP\* | Адрес первого элемента первой структуры массива исходных данных |
| int | len | Размер массива |
| **Формальные параметры** | ships | int\* | Указатель на переменную, хранящую размер массива |
| **Локальные переменные** | is\_end\_input | bool | Вспомогательная переменная |
| ship | SHIP\* | Адрес первого элемента первой структуры массива исходных данных |

# 9.14 Функция processing

**Назначение функции:**

Это функция для обработки данных.

**Прототип:**

SHIP \*processing(SHIP \*ship, int ship\_len, int \*ship\_result\_len);

**Возвращаемое значение:**

Эта функция возвращает .

**Пример вызова:**

ship\_result = processing(ship,ship\_len,&ship\_result\_len);

**Описание переменных:**

Описание переменных в таблице 17. Переменные функции processing

Таблица 17. Переменные функции processing

|  |  |  |  |
| --- | --- | --- | --- |
| **Вид параметра** | **Имя**  **переменной** | **Тип переменной** | **Назначение переменной** |
| **Фактические параметры** | ship\_result | SHIP\* | Адрес первого элемента первой структуры массива результирующих данных |
| ship | SHIP\* | Адрес первого элемента первой структуры массива исходных данных |
| ship\_len | int | Размер массива исходных данных |
| ship\_result\_len | int | Размер массива результирующих данных |
| **Формальные параметры** | ship | SHIP\* | Адрес первого элемента первой структуры массива исходных данных |
| ship\_len | int | Размер массива исходных данных |
| ship\_result\_len | int\* | Указатель на переменную, в которой хранится размер массива результирующих данных |
| **Локальные переменные** | i | int | Вспомогательная переменная |
| r | int | Вспомогательная переменная |
| len | int | Длинна строки |
| name\_find | char\* | Указатель на переменную, в которой хранится требуемое имя |
| type\_find | char\* | Указатель на переменную, в которой хранится требуемый тип |
| is\_type\_name | bool | Флажок поиска по имени |
| is\_type\_type | bool | Флажок поиска по типу |

# 10.Текст программы

#include <conio.h>

#include <malloc.h>

#include <stdio.h>

#include <windows.h>

#include <locale.h>

#include <math.h>

typedef struct ST

{

char \*name;//Указатель на первый символ строки, содержащий имя кораблика

char \*type;//Указатель на первый символ строки, содержащий тип кораблика

int x; //Положение кораблика по x

int y; //Положение кораблика по y

int z; //Положение кораблика по z

} SHIP;

//Функция для получения символа

char my\_getch()

{

char ch;//Код нажатой кнопки

do

{

ch = \_getch();

if (ch == '\r')

ch = '\n';

else

if (ch == '\0' || ch == -32)

\_getch();

else

if (ch == '\t')

ch = ' ';

}

while (ch >= -128 && ch <= -17 || ch == '\0' || ch == -32);

return ch;

}

//Функция для ввода целого числа

int input\_number()

{

int number; //Введённое число

char \*s; //Указатель на первый символ исходной строки

int length = 1;//Длина строки

char ch; //Введённый символ

s = NULL;

do

{

ch = my\_getch();

switch (ch)

{

case 27:

break;

case '\b':

if (length > 1)

{

printf("\b \b");

length--;

s = (char \*)realloc(s, sizeof(char)\*length);

}

break;

case '\n':

if (length>1)

s[length - 1] = 0;

break;

default:

if ((ch >= '0' && ch <= '9' || ch == '-' && length == 1) && length <= 7)

{

s = (char \*)realloc(s, sizeof(char)\*(length + 1));

printf("%c", ch);

s[length - 1] = ch;

length++;

}

break;

}

}

while (ch != '\n' || length == 1 || length == 2 && \*s == '-');

number = atoi(s);

free(s);

printf("\n");

return number;

}

//Функция для ввода ответа y/n

bool input\_yn()

{

char sy; //Введённый символ

char ch; //Нажатая клавиша

sy = 0;

do

{

ch = my\_getch();

switch (ch)

{

case '\b':

printf("\b \b");

sy = 0;

break;

case 'y':

case 'Y':

case 'n':

case 'N':

sy = ch;

printf("\b%c", sy);

break;

}

}

while (ch != '\n' || sy == 0);

printf("\n");

return sy == 'y' || sy == 'Y';

}

//Функция для ввода строки

char \*input\_string(int size)

{

char \*s; //Указатель на первый символ исходной строки

int length;//Длина строки

char ch; //Введённый символ

length = 1;

s = NULL;

do

{

ch = my\_getch();

switch (ch)

{

case 27:

break;

case '\b':

if (length > 1)

{

printf("\b \b");

length--;

s = (char \*)realloc(s, sizeof(char)\*length);

}

break;

case '\n':

if (length > 1)

{

while (s[length - 2] == ' ')

{

length--;

s = (char \*)realloc(s, sizeof(char)\*length);

}

s[length - 1] = 0;

}

break;

default:

if (length <= size && (ch != ' ' || length > 1))

{

s = (char \*)realloc(s, sizeof(char)\*(length + 1));

printf("%c", ch);

s[length - 1] = ch;

length++;

}

break;

}

}

while (ch != '\n' || length == 1);

printf("\n");

return s;

}

//Функция для вычисления длинны строки

int str\_len(char\*s)

{

int i = 0;//Длина строки

while (s[i])

i++;

return i+1;

}

//Функция для очистки памяти из под структур

SHIP \*free\_structs(SHIP \*ship, int len)

{

int i;//Вспомогательная переменная

if (ship != NULL)

{

for (i = 0; i < len; i++)

{

free(ship[i].name);

free(ship[i].type);

}

free(ship);

}

return NULL;

}

//Функция для вывода данных одной структуры

void print\_struct(SHIP ship)

{

printf("Имя кораблика: \"%s\"\n", ship.name);

printf("Тип кораблика: \"%s\"\n", ship.type);

printf("Положение кораблика по x:%i\n", ship.x);

printf("Положение кораблика по y:%i\n", ship.y);

printf("Положение кораблика по z:%i\n", ship.z);

}

//Функция для вывода всех структур

void print\_structs(SHIP \*ship,int ships)

{

int i;//Вспомогательная переменная

if (ship != NULL)

{

for (i = 0; i < ships; i++)

{

if (i % 3 == 0 && i > 0)

system("pause");

printf("\nКораблик %i\n", i);

print\_struct(ship[i]);

}

puts("\nВывод окончен");

}

else

puts("Массив пуст");

}

//Функция для ввода типа

char\* get\_type()

{

char \*types[4] = { "battleship", "cruiser", "boat", "steelship" };

//Массив указателей на первый символ строки

int i; //Вспомогательная переменная

char \*s; //Указатель на первый символ вводимой строки

char type;//Тип ораблика

char len; //Длина строки

puts("Выберите тип:");

for (i = 0; i < 4; i++)

printf("%i-%s\n", i + 1, types[i]);

puts("другое число-ввод типа вручную\n");

type = input\_number();

if (type <= 0 || type >= 5)

{

puts("Вводите тип (от 1 до 20 символов):");

s = input\_string(20);

}

else

{

len = str\_len(types[type - 1]);

s = (char\*)malloc(sizeof(char)\*len);

for (i = 0; i < len; i++)

s[i] = types[type - 1][i];

}

return s;

}

//Функция для ввода имени

char\* get\_name()

{

char \*s; //Указатель на первый символ строки

bool exit;//Флажок выхода

exit = 0;

do

{

puts("Введите имя (имя должно содержать хотя бы один символ");

puts("и не более 30 символов):");

s = input\_string(30);

if (\*s < 'A' || \*s > 'Z')

{

puts("Вниание! Имя корабля начинается не с большой буквы.");

puts("Вы хотите ввести другое имя? (y/n)");

if (input\_yn())

free(s);

else

exit = 1;

}

else

exit = 1;

}

while (!exit);

return s;

}

//Функция для ввода координат

int get\_coord(char ch)

{

printf("Вводите положение по %c в сантиметрах (целое число от -999999 до

9999999):\n",ch);

return input\_number();

}

//Функция для ввода полей структуры

void get\_struct(SHIP \*info)

{

info->name = get\_name();

info->type = get\_type();

info->x = get\_coord('x');

info->y = get\_coord('y');

info->z = get\_coord('z');

}

//Функция для ввода массива структур

SHIP \*get\_structs(int \*ships)

{

bool is\_end\_input;//Вспомогательная переменная

SHIP \*ship; //Адресс первого элемента первой структуры массива исходных данных

is\_end\_input = false;

ship = NULL;

\*ships = 0;

do

{

ship = (SHIP\*)realloc(ship, (\*ships + 1)\*sizeof(SHIP));

get\_struct(ship + \*ships);

(\*ships)++;

puts("Продолжить ввод?(y/n)");

is\_end\_input = !input\_yn();

}

while (!is\_end\_input);

return ship;

}

//Функция для обработки данных

SHIP \*processing(SHIP \*ship, int ship\_len, int \*ship\_result\_len)

{

SHIP \*ship\_result;//Адресс

int i; //Вспомогательная переменная

int r; //Вспомогательная переменная

int len; //Длинна строки

char \*name\_find; //Требуемое имя

char \*type\_find; //Требуемый тип

bool is\_type\_name;//Флажок поиска по имени

bool is\_type\_type;//Флажок поиска по типу

do

{

puts("Выберите критерий:");

puts("1-по имени");

puts("2-по типу");

puts("3-по типу и по имени");

i = input\_number();

system("cls");

if (i < 1 || i > 3)

{

puts("Нет такого пункта");

system("pause");

system("cls");

}

}

while (i < 1 || i > 3);

is\_type\_name = i == 1 || i == 3 ? 1 : 0;

is\_type\_type = i == 2 || i == 3 ? 1 : 0;

if (is\_type\_name)

name\_find = get\_name();

if (is\_type\_type)

type\_find = get\_type();

ship\_result = NULL;

\*ship\_result\_len = 0;

for (i = 0; i<ship\_len; i++)

if ((!is\_type\_name || !strcmp(name\_find, ship[i].name))

&& (!is\_type\_type || !strcmp(type\_find, ship[i].type)))

{

ship\_result = (SHIP\*)realloc(ship\_result, sizeof(SHIP)\*(\*ship\_result\_len + 1));

len = str\_len(ship[i].name);

(ship\_result + \*ship\_result\_len)->name = (char\*)malloc(sizeof(char)\*len);

for (r = 0; r < len; r++)

(ship\_result + \*ship\_result\_len)->name[r] = ship[i].name[r];

len = str\_len(ship[i].type);

(ship\_result + \*ship\_result\_len)->type = (char\*)malloc(sizeof(char)\*len);

for (r = 0; r < len; r++)

(ship\_result + \*ship\_result\_len)->type[r] = ship[i].type[r];

(ship\_result + \*ship\_result\_len)->x = ship[i].x;

(ship\_result + \*ship\_result\_len)->y = ship[i].y;

(ship\_result + \*ship\_result\_len)->z = ship[i].z;

(\*ship\_result\_len)++;

}

if (is\_type\_name)

free(name\_find);

if (is\_type\_type)

free(type\_find);

return ship\_result;

}

int main()

{

SHIP \*ship=NULL; //Указатель на первый элемент исходного массива

SHIP \*ship\_result=NULL; //Указатель на первый элемент обработанного массива

int ship\_len; //Количество элементов в исходном массиве

int ship\_result\_len; //Количество элементов в обработаном массиве

int menu; //Выбраный пункт меню

bool is\_input = false; //Флажок наличия исходного массива (1-исходный массив

введён, 0-исходный массив не введён)

bool is\_processed = false;//Флажок проведения обработки массива (1-обработка была, 0-

обработки не было)

bool is\_output = false; //Флажок наличия исходного массива (1-исходный массив

введён, 0-исходный массив не введён)

setlocale(0, "RU");

do

{

system("cls");

puts("Меню");

puts("1-Ввести данные о корабликах");

puts("2-Вывести данные о корабликах");

puts("3-Обработать данные о корабликах");

puts("4-Вывести обработанные данные");

puts("5-Справка");

puts("0-Выход");

menu = input\_number();

system("cls");

switch (menu)

{

case 1:

ship=free\_structs(ship, ship\_len);

ship\_result=free\_structs(ship\_result, ship\_result\_len);

is\_input = true;

is\_output = false;

is\_processed = false;

ship = get\_structs(&ship\_len);

puts("\nВвод окончен");

system("pause");

break;

case 2:

if (is\_input)

print\_structs(ship, ship\_len);

else

puts("Данные ещё не введены");

system("pause");

break;

case 3:

if (is\_input)

{

ship\_result=free\_structs(ship\_result, ship\_result\_len);

ship\_result = processing(ship,ship\_len,&ship\_result\_len);

if (ship\_result\_len > 0)

{

is\_output = true;

puts("Обработка успешно завершена");

}

else

{

is\_output = false;

puts("Не найдено корабликов, соответствующих данному критерию");

}

is\_processed = true;

}

else

puts("Данные не введены");

system("pause");

break;

case 4:

if (is\_input)

if (is\_processed)

if (is\_output)

print\_structs(ship\_result, ship\_result\_len);

else

printf("Не найдено корабликов, соответствующих данному критерию");

else

printf("Обработка не проводилась");

else

printf("Нет входных данных");

printf("\n");

system("pause");

break;

case 5:

puts("Эта программа обрабатывает заданный массив данных о корабликах.");

puts("Кораблик имеет пять полей:");

puts("1. Имя-строка от 1 до 30 символов");

puts("2. Тип-строка от 1 до 20 символов");

puts("3. Положение по x-целое число от -999999 до 9999999");

puts("4. Положение по y-целое число от -999999 до 9999999");

puts("5. Положение по z-целое число от -999999 до 9999999");

puts("Результат обработки - новый массив данных состоящий из элементов

исходного");

puts("массива, которые соответствуют заданным критериям.");

puts("Русские буквы вводить запрещено.");

system("pause");

break;

case 0:

break;

default:

printf("Неправильно введён пункт меню\n");

system("pause");

break;

}

}

while (menu != 0);

ship=free\_structs(ship, ship\_len);

ship\_result=free\_structs(ship\_result, ship\_result\_len);

}

# 11.Результат

При выполнении программы получены результаты, совпадающие с контрольными примерами, представленными в таблице 1. Контрольные примеры. Ошибки не обнаружены.

# Вывод

В результате выполнения лабораторной работы получены практические навыки работы с структурами в языке C/C++.