ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| Лабораторная работа 3. Программирование поразрядных операций |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4134К |  |  |  | Иванов И. В. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

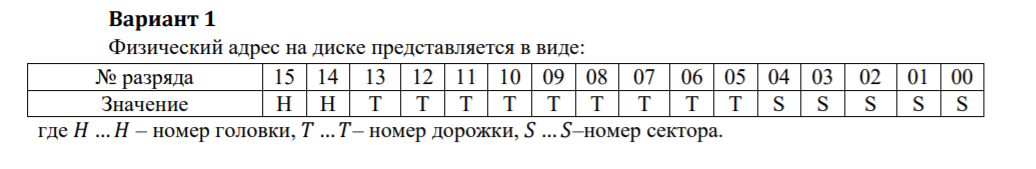
Санкт-Петербург 2022

**Цель работы:**

Целью работы является изучение поразрядных операций типа НЕ, И, ИЛИ, исключающее ИЛИ и операций сдвига.

**Задание на лабораторную работу**:

Составить две программы, первая из которых вводит составные части структуры данных, приведённой в индивидуальном варианте, как десятичные числа и формирует из них заданную упакованную структуру как 16-ричное число. Вторая программа вводит упакованную структуру как 16-ричное число и выводит значения отдельных её составных частей как десятичные числа.



**Ход выполнения:**

**Использованные функции:**

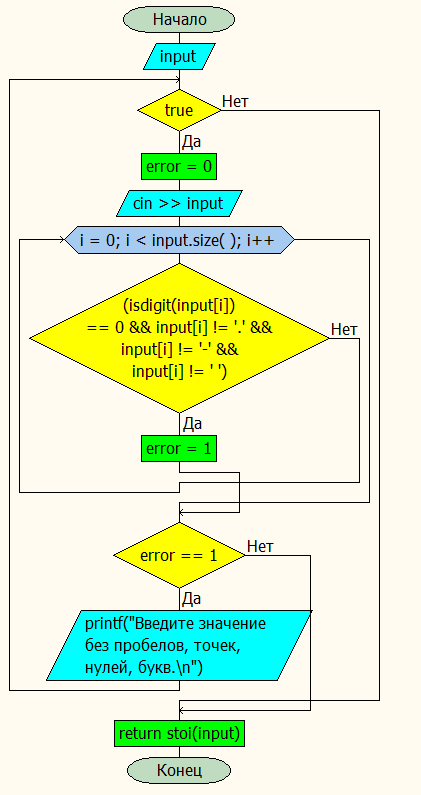
**Имя:** proverka  
**Назначение:** улучшенный ввод с клавиатуры, Который запрашивает строку, проверяет ее на возможность перевода в int

**Входные данные:** Запрос с клавиатуры

**Выходные данные:** Ввод с клавиатуры, преобразованный в int

**Побочные эффекты:** Отсутствует

**Блок схема:**

****

**Псевдокод:**

Ввод с клавиатуры

Пока 1

Проверка каждого элемента введенной строки

Если элемент не подходит условиям isdigit()

Ошибка=1

Если ошибки есть, просим ввести заново

Иначе конец

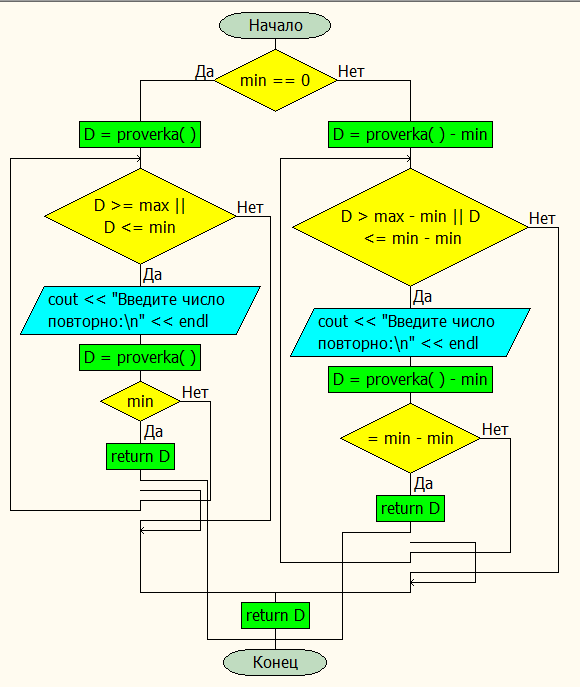
**Имя:** check\_input  
**Назначение:** проверка правильности границ вводимого числа

**Входные данные:** значение, возвещаемое из функции proverka

**Выходные данные:** значение, прошедшее проверку правильности границ

**Побочные эффекты:** Отсутствует

**Блок-схема:**

****

**Псевдокод:**

Объявления переменной D

Если минимальное значение границы равно 0, то выполняем if. D = проверенному числу из функции proverka.

Если число не удовлетворяет условиям цикла, то выводим D, если удовлетворяет, то выполняем цикл.

Если число удовлетворяет условиям цикла, то просим ввести число повторно и начинаем цикл заново, если нет – выводим число.

Если минимальное значение границы не равно 0, то выполняем else. D = проверенному числу из функции proverka, минус min.

Если число не удовлетворяет условиям цикла, то выводим D, если удовлетворяет, то выполняем цикл.

Если число удовлетворяет условиям цикла, то просим ввести число повторно и начинаем цикл заново, если нет – выводим число.

Вывод D

**Имя:** main()

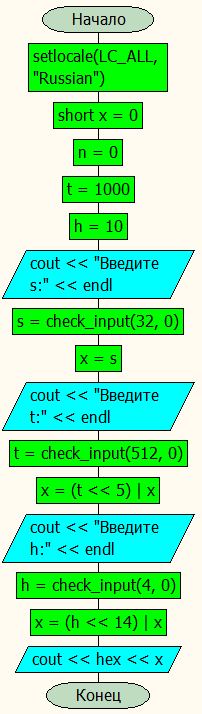
**Назначение:** запаковать данные согласно варианту в одну ячейку памяти

**Входные данные:** 3 ввода с клавиатуры

**Выходные данные:** ячейка памяти, содержащая в себе 3 переменные

**Побочные эффекты:** Шестнадцатеричный формат

**Блок-схема:**

****

|  |  |
| --- | --- |
| **Входные данные** | **Выходные данные** |
| **30 356 2** | **ac9e** |
| **26 245 3** | **deba** |

**Псевдокод:**

Объявления 4 переменных

Ввод 3 переменных с клавитуры с параллельной проверкой область допостимых значений

Упаковка всех введенных значений в X

Вывод X

**Имя:** raspakovka()

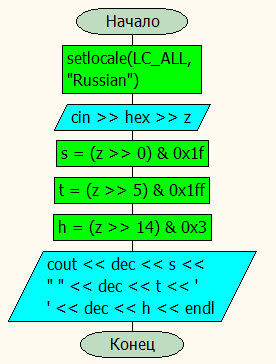
**Назначение:** распаковать данные согласно варианту, из одной ячейки памяти

**Входные данные:** 1 ввод с клавиатуры

**Выходные данные:** 3 переменные, восстановленные из упаковки

**Побочные эффекты:** нет

**Блок-схема:**

****

|  |  |
| --- | --- |
| **Входные данные** | **Выходные данные** |
| **ac9e** | **30 356 2** |
| **deba** | **26 245 3** |

**Псевдокод:**

Ввод упаковки с клавиатуры

Пошаговое извлечение каждого элемента из ячейки памяти

Их вывод

**Листинг всей программы:**

#include <iostream>

#include <string>

#include <bitset>

using namespace std;

int proverka() {

string input;

while (true) //проверка введенного числа

{

bool error = 0;

cin >> input;

for (int i = 0; i < input.size(); i++)

{

if ((isdigit(input[i]) == 0 && input[i] != '.' && input[i] != '-' && input[i] != ' ')) {

error = 1;

break;

}

}

if (error == 1) {

printf("Введите значение без пробелов, точек, нулей, букв.\n");

}

else

{

break;

}

}

return stoi(input);

}int check\_input(int max, int min) {

int D;

if (min == 0) {

D = proverka();

while (D >= max || D <= min) { //ввод приемлемых значений

cout << "Введите число повторно:\n" << endl;

D = proverka();

if (D <= max && D > min) {

return D;

break;

}

}

}

else {

D = proverka() - min;

while (D > max - min || D <= min - min) { //ввод приемлемых значений

cout << "Введите число повторно:\n" << endl;

D = proverka() - min;

if (D <= max - min && D >= min - min) {

return D;

break;

}

}

}

return D;

}

int main() {

setlocale(LC\_ALL, "Russian");

unsigned short x = 0;

unsigned short s;

unsigned short h;

unsigned short t;

int n;

n = 0;

t = 1000;

h = 10;

cout << "Введите s:" << endl;

s = check\_input(32, 0);

x = s;

cout << "Введите t:" << endl;

t = check\_input(512, 0);

x=( t << 5 ) | x;

cout << "Введите h:" << endl;

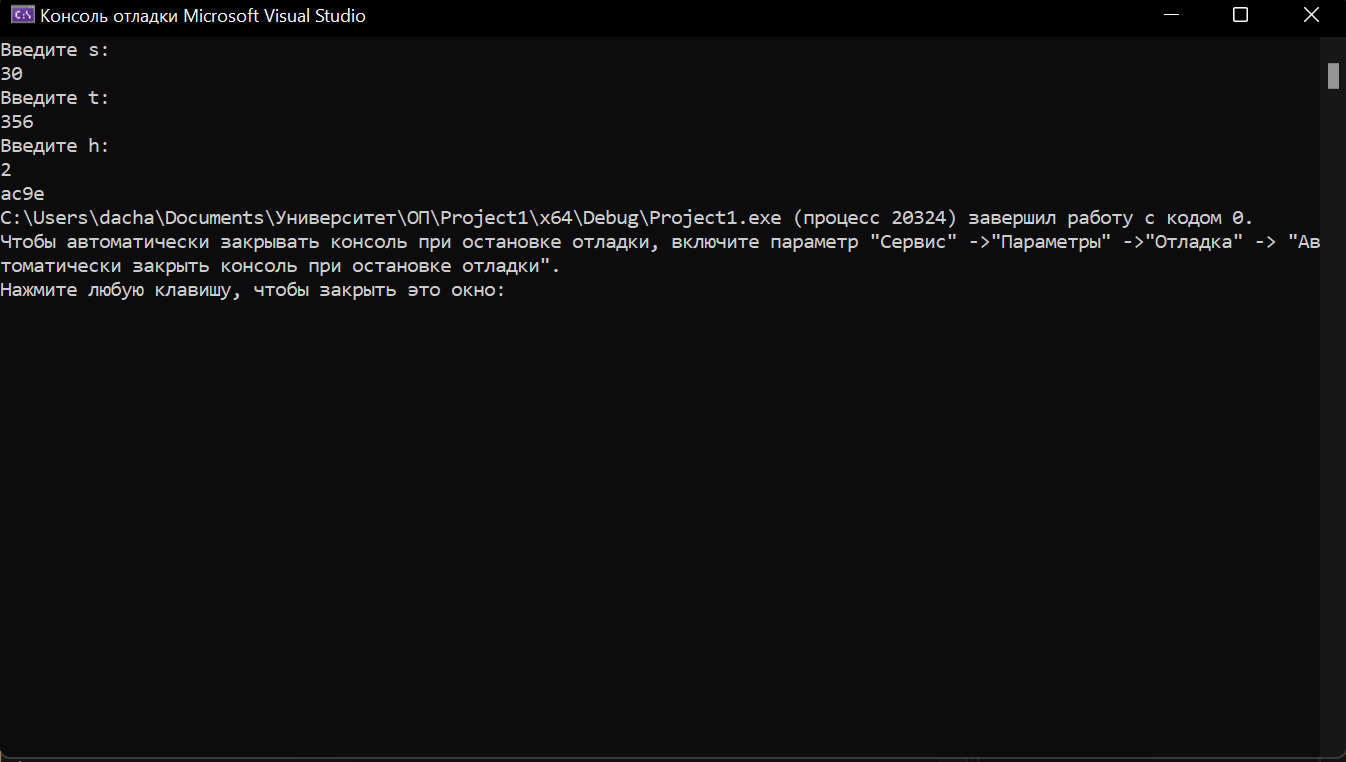
h = check\_input(4, 0);

x = (h << 14) | x;

cout << hex << x;

}

**Результат работы программы:**

****

**Листинг программы raspakovka:**

#include<iostream>

#include <bitset>

using namespace std;

int main() {

setlocale(LC\_ALL, "Russian");

unsigned short z;

short s, t, h;

cin >> hex >> z;

s = (z >> 0) & 0x1f;

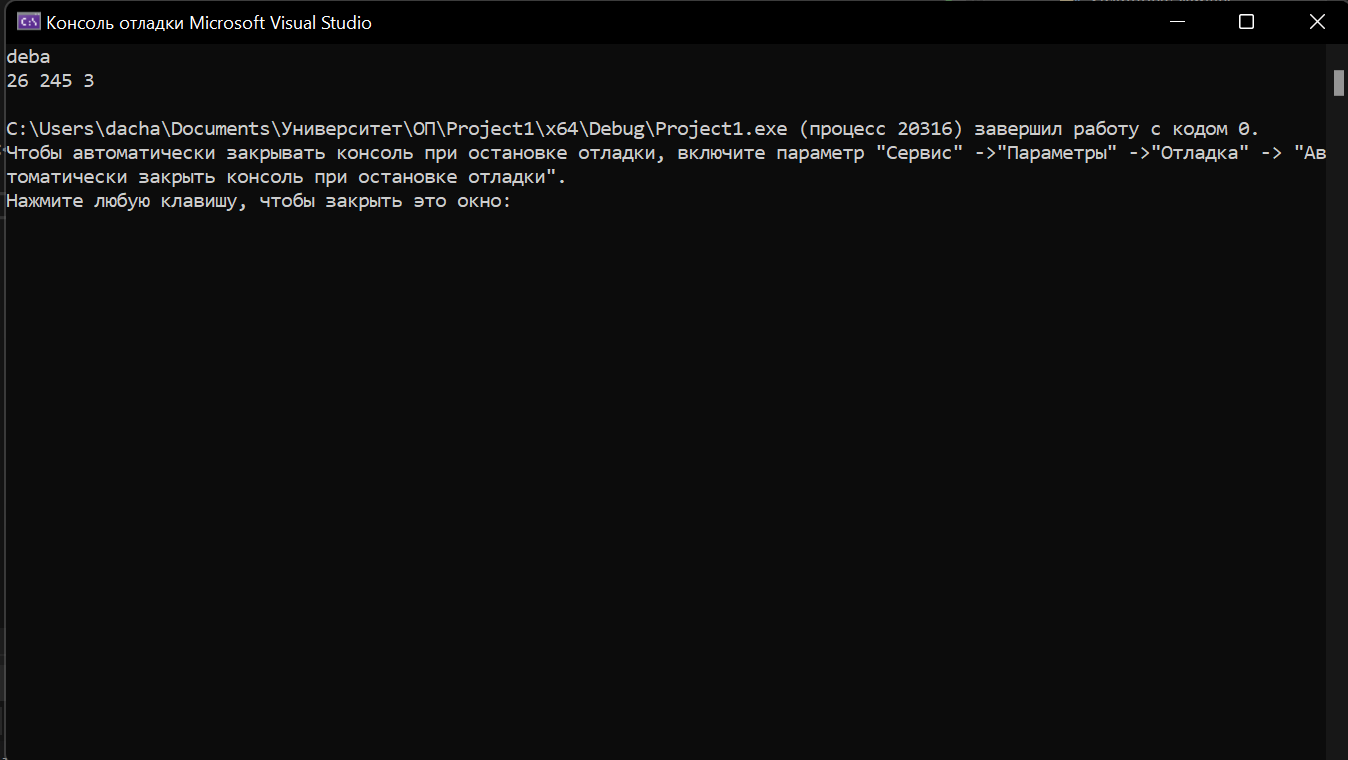
t = (z >> 5) & 0x1ff;

h = (z >> 14) & 0x3;

cout << dec << s << " " << dec << t << ' ' << dec << h << endl;

}

**Результат работы raspakovka:**

****

**Вывод:**

Я изучил операции сдвига и смог запаковать 3 переменные в одну ячейку памяти, а потом распаковать ее.

**Достоинства программы:**

Скорость работы.

Проверка значений.

Наличие пользовательского интерфейса.

**Недостатки программы:**

Недостатков нет.