|  |
| --- |
| **МИНОБРНАУКИ РОССИИ**  **САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  **ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  **«ЛЭТИ» ИМ. В. И. УЛЬЯНОВА (ЛЕНИНА)**  **Кафедра РЭС** |
| **ОТЧЕТ по лабораторной работе №3 по дисциплине «Информатика» Тема: «Автоматизация перевода чисел из десятичной системы счисления в двоичную с использованием поразрядных операций и фильтрацией входных данных»** |
| |  |  | | --- | --- | | Студент гр. 1183 | Чаминов Д. А. | | Преподаватель | Ситников И. Ю. | |
| Санкт-Петербург  2021 |

**Содержание**

[Спецификация задания 3](#_Toc85750088)

[Описание алгоритма 4](#_Toc85750089)

[Блок-схема алгоритма 5](#_Toc85750090)

[Выбор и обоснование типов переменных 9](#_Toc85750091)

[Вводимые и выводимые параметры и их типы 10](#_Toc85750092)

[Структура проекта 11](#_Toc85750093)

[Текст программы 12](#_Toc85750094)

[Копии экрана при работе программы 16](#_Toc85750095)

[Контрольный пример, сравнение с эталоном 17](#_Toc85750096)

[Результаты 19](#_Toc85750097)

# Спецификация задания

Разрабатываемое приложение должно отвечать следующим требованиям:

* Работать в операционных системах MS Windows 7 и выше;
* Разрабатываться в среде MS Visual Studio на языке C (Си);
* Принимать на вход не более 100 строк, содержащих различные символы, осуществлять фильтрацию этих строк (игнорировать символы, не являющиеся цифрами или знаком исходного числа), выводить 32 двоичных разряда (с использованием дополнительного двоичного кода) для каждого из введенных чисел (вывод должен осуществляться в виде таблицы);
* Быть создано с использованием разработанной библиотеки, содержащей основные функции, необходимые для выполнения задачи.

# Описание алгоритма

В программу подаются целые десятичные числа, которые она сохраняет в массиве. После завершения ввода и подтверждения выхода для каждого из исходных чисел вызывается функция represent\_as\_bin, которая при помощи битовых операторов циклически вычисляет каждый бит числа (для проверки бита с номером i необходимо проверить результат поразрядной конъюнкции для исходного числа и : если результат операции равен 0, то этот бит – 0, иначе – 1), записывает двоичное представление числа в переданную по указателю си-строку, которая выводится вместе с исходным числом и номером запроса.

# Блок-схема алгоритма

Основной алгоритм:

char[33] res;

represent\_as\_bin(res, request[i]);

Вывод i, request[i], res

нетt

да

i++;

i=0;

requests[i]==0?

да

нет

did\_user\_accept()

да

нет

requests[i]=tmp;

i++;

int requests[100]={ 0 };

int tmp, i=0;

tmp==0?

Ввод str

filter(str);

tmp=atoi(str);

Алгоритм filter:

да

да

нетt

нетt

str[j++]=str[i];

i++;

str[i]<=’9’ && str[i] >=’0’ || str[i]==’-‘ && j==0 ?

str[i]==0?

i=j=0;

Получение параметра str

Алгоритм did\_user\_accept:

да

да

Вернуть «да»

Вернуть «нет»

нетt

нетt

a совпадает с “n”?

a совпадает с “y”?

Вывод “Are you sure (y/n): ”  
Ввод строки a

Алгоритм represent\_as\_bin:

int i =0;

Получение параметров char\* result, int a

result[32]=’\0’

нет

i++

нет

да

result[31-i]=’0’

result[31-i]=’1’

a&(1<<i)!=0

да

i<32

# Выбор и обоснование типов переменных

Так как программа должна выводит 32 разряда числа (с использованием дополнительного кода), то для этого будет достаточно 32-битного знакового целого числа (int). Для вывода результата удобнее всего использовать си-строку – массив символов (char [33]) 32 символа достаточно для двоичного представления числа, и еще 1 символ – конец строки ‘\0’. Поскольку количество исходных чисел не превышает 100, а каждое из них является целым и принадлежит промежутку   
, то для их хранения будет достаточно int [100]. Строки, используемые в программе, вводятся пользователем. Из расчета, что стандартная ширина окна консоли составляет 80 символов, то с небольшим запасом для их представления будет достаточно char[100]. Это значение задается макросом MY\_STR\_SIZE и может быть легко изменен при надобности.

# Вводимые и выводимые параметры и их типы

Перед вводом пользователю выводится приглашение (const char[23]) для первого числа и в случае отмены выхода, const char[3] для всех остальных чисел). После окончания ввода программа запрашивает подтверждение окончания ввода (const char[21]). После подтверждения завершения ввода программа выводит шапку таблицы (const char[52]), а затем для каждого введенного пользователем числа (кроме 0, являющегося сигналом конца ввода) программа выводит форматированную строку из 3х символов (выравнивание по центру), разделитель из 3х символов, форматированную строку (14 символов, по правому краю), содержащую исходное число, разделитель из 3х символов, двоичное представление числа (char[33]), и перевод каретки.

На ввод программе подаются строки, длина которых не превышает MY\_STR\_SIZE=100 символов, каждая из которых содержит в себе число в промежутке (строки проходят фильтрацию, поэтому все символы, не являющиеся знаком числа или цифрой будут игнорироваться). Количество строк не должно превышать 100, для завершения ввода следует ввести “0”. Затем пользователь вводит подтверждение – строку char[MY\_STR\_SIZE=100], которая должна быть “y” (в случае если вы действительно хотели завершить ввод) или “n” (если вы хотите продолжить ввод чисел). В случае, если введенная строка не совпадает с вышеперечисленными, подтверждение выхода будет запрошено еще раз.

# Структура проекта

Папка проекта содержит 4 файла:  
21.10.2021 00:02 2 249 Lab3.cpp

20.10.2021 12:44 7 344 Lab3.vcxproj

20.10.2021 12:44 1 336 Lab3.vcxproj.filters

20.10.2021 12:43 168 Lab3.vcxproj.user

Также в проекте задействованы 2 файла из другой директории:

21.10.2021 00:20 3 075 myDecToBinLibFunctions.cpp

21.10.2021 00:00 1 438 myDecToBinLibFunctions.h

# Текст программы

// Файл Lab3.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS // MS VS forbids unsafe functions like 'scanf'

#include <stdio.h> // standard input/output library

#include "../libs/myDecToBinLibFunctions.h" // my library with my functions

#include <stdlib.h> // for atoi and itoa functions

int main()

{

char A[MY\_STR\_SIZE] = { 0 }; // string for input

int requests[100] = { 0 }; // array for initial numbers

int i = 0; // just a counter

do {

printf("Input your numbers\n>>>"); //prompt for input

scanf("%s", A); // get string

int tmp = atoi(filter\_string(A)); //filter string and get number from it

while (tmp) { //while inputed not zero

requests[i++] = tmp; // put this request into array

printf(">>>"); // prompt for input

scanf("%s", A); // get string

tmp = atoi(filter\_string(A)); //filter string and get number from it

}

} while (!did\_user\_accept()); //while user doesn’t accept exiting

printf("Output:\n # | decimal | binary\n"); // table header

i = 0;

while (requests[i]) { // for each inputted number

char res[33]; // c-string for binary representation

represent\_as\_bin(requests[i], res); // represent as bin and write

// representation in res

//printf("%-5d%-+16d%s\n", i + 1, requests[i], res); // print row of the table

char number\_s[4] = { 0 },

decimal\_s[15] = { 0 };

\_itoa(i+1, number\_s, 10); // make string with index + 1

\_itoa(requests[i], decimal\_s, 10); // make string with requested number

inflate\_string(number\_s, 3, StringAlign::center); // format string

inflate\_string(decimal\_s, 14, StringAlign::right); //format string

char row[80] = { 0 }; // string for the row of the table

strcat(row, number\_s); // add index + 1 string to the row

strcat(row, " | "); // add separator to the row

strcat(row, decimal\_s); // add requested number string to the row

strcat(row, " | "); // add separator to the row

strcat(row, res); // add binary representation to the row

printf("%s\n", row); // print row

i++; // increase index

}

do\_pause(); // do not close console, user should be able to see result

return 0; // EXIT\_SUCCESS

} //Файл myDecToBinLibFunctions.h

#ifndef myDecToBinLibFunctionsINCLUDED

#define myDecToBinLibFunctionsINCLUDED

#define MY\_STR\_SIZE 100 // size of the string, that user will be input

#define \_CRT\_SECURE\_NO\_WARNINGS // MS VS do forbids unsafe functions like 'scanf'

#include <stdbool.h> // standard bool type library

#include <string.h> // standard string functions library

#include <stdio.h> // standard input/output library

#include <process.h> // library for system functions

char\* filter\_string(char\* s);

/\*

removes all 'bad' characters (not digits or minus if it is first)

in the c-string

function is protected by limitation string size with MY\_STR\_SIZE

\*/

void represent\_as\_bin(int n, char\* b);

/\*

Represents -2^16 <= n < 2^16 in binary notation and writes

result to the c-string b, that MUST BE AT LEAST 32 SYBLOS LENGTH

\*/

bool did\_user\_accept();

/\*

Asks user if he really want to exit

returns true if it's true else returns false

\*/

void do\_pause(char print\_dir = 0);

/\*

Do not close console,

if print\_dir is not 0 - uses dir command

function is safe

\*/

enum StringAlign { left, center, right }; // types of the string align for inflate\_string function

char\* inflate\_string(char\* s, unsigned int width, StringAlign align = StringAlign::left, char filler = ' ');

/\*

Increases string size with adding filler symbols according to align parameter

\*/

#endif

//Файл myDecToBinLibFunctions.cpp

#include "myDecToBinLibFunctions.h"

char\* filter\_string(char\* s)

{ /\*

removes all 'bad' characters (not digits or minus if it is first)

in the c-string

function is protected by limitation string size with MY\_STR\_SIZE

\*/

int i, // index of the char that we read from the string

j; // index of the char that we write to the string

for (i = j = 0; i < MY\_STR\_SIZE && s[i] != '\0'; i++) { // for each char in the string

if (('0' <= s[i] && s[i] <= '9') || (j == 0 && s[i] == '-')) {// if char is not bad

s[j++] = s[i]; // we write it

}

}

s[j] = '\0'; // end of c-string

return s;

}

void represent\_as\_bin(int n, char\* b)

{

/\*

Represents -2^16 <= n < 2^16 in binary notation and writes

result to the c-string b, that MUST BE AT LEAST 32 SYBLOS LENGTH

\*/

int i; // bit index

for (i = 0; i < 32; i++) { // i reaches from 0 to 32 increasing by 1

if ((n & (1 << (31 - i)))) { // if this bit is 1

// ^ use mask

// ^ set mask to 0...1...0, where 1 is at (31-i) bit

b[i] = '1'; //set current char in c-string to '1'

} else {

b[i] = '0'; // set current char in c-string to '0'

}

}

b[i] = '\0'; // end of c-string

}

void do\_pause(char print\_dir)

{

/\*

Do not close console,

if print\_dir is not 0 - uses dir command

function is safe

\*/

if (system(NULL)) {

if (print\_dir) {

system("DIR");

}

system("pause");

}

}

char\* inflate\_string(char\* s, unsigned int width, StringAlign align, char filler)

{

/\*

Increases string size with adding filler symbols according to align parameter

\*/

unsigned int n = strlen(s); // get length

switch (align) { // for the different aligns

case StringAlign::left:

{

int i;

for (i = n; i < width; i++) {

s[i] = filler; // just add fillers after end of

// string until thehy reach width

}

break;

}

case StringAlign::right:

{

int i;

for (i = width - 1; i >= 0; i--) {

s[i] = width - i <= n ? s[n - (width - i)] : filler;

// copy symbols to the end string if we have symbols left

// else just put fillers

}

break;

}

case StringAlign::center:

{

inflate\_string(s, (width - n) / 2 + n, StringAlign::right, filler);

// add fillers to the left

inflate\_string(s, width, StringAlign::left, filler);

// add fillers to the right

break;

}

}

s[width] = '\0'; // end of string

return s;

}

bool did\_user\_accept()

{ /\*

Asks user if he really want to exit

returns true if it's true else returns false

\*/

char a[MY\_STR\_SIZE] = "a"; // c-string for answer

do {

printf("Are you sure (y/n): "); //prompt for input

scanf("%s", a); // get answer

// while answer is neither 'y' nor 'n'

} while (!(!strncmp(a, "y", MY\_STR\_SIZE) || !strncmp(a, "n", MY\_STR\_SIZE)));

return !strcmp(a, "y"); // return if user want to exit

}

# Копии экрана при работе программы

Изображение выглядит как текст

Автоматически созданное описание Изображение выглядит как текст

Автоматически созданное описание

# Контрольный пример, сравнение с эталоном

Для проверки алгоритма будут использоваться следующие входные данные:

qwe2qwe0qwe2qwe1

qwe-qwe2qwe0qwe2qwe1

qwe2qwe-qwe1

0

m

n

-1

7

15

-15

0  
y

Ожидаемый результат:

00000000000000000000011111100101

11111111111111111111100000011011

00000000000000000000000000010101

11111111111111111111111111111111

00000000000000000000000000000111

00000000000000000000000000001111

11111111111111111111111111110001

Полученный результат:  
 Изображение выглядит как текст

Автоматически созданное описание

# Результаты

* Разработанная программа работает на операционных системах MS Windows 7 и выше.
* Приложение является консольным.
* Файл Lab3.exe имеет размер 11 кб.
* Программа работает с не более чем 100 строками, содержащими целые числа (последовательности всех символов кроме цифр и знака «-» игнорируются) в промежутке .
* Проверка показала корректность работы алгоритма.
* Достоинства:
  + Быстрый (по сравнению с алгоритмом деления) перевод чисел;
  + Возможность работать с несколькими числами не перезапуская программу;
  + Удобный формат вывода;
  + Защита от случайных нажатий клавиш;
  + Использование библиотеки, что дает возможность для повторного использования кода, его поддержки;
  + Понятный интерфейс.
* Недостатки:
  + Ограничения на длину вводимых строк, их количество, диапазон чисел;
  + Интерфейс программы доступен только на английском языке.