**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по лабораторной работе №1**

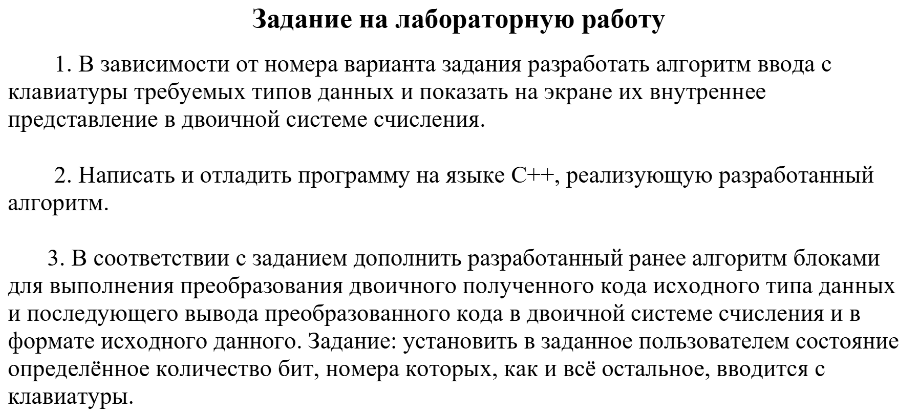
**по дисциплине «ОЭВМ»**

# Тема: **ИССЛЕДОВАНИЕ ВНУТРЕННЕГО ПРЕДСТАВЛЕНИЯ РАЗЛИЧНЫХ ФОРМАТОВ ДАННЫХ**

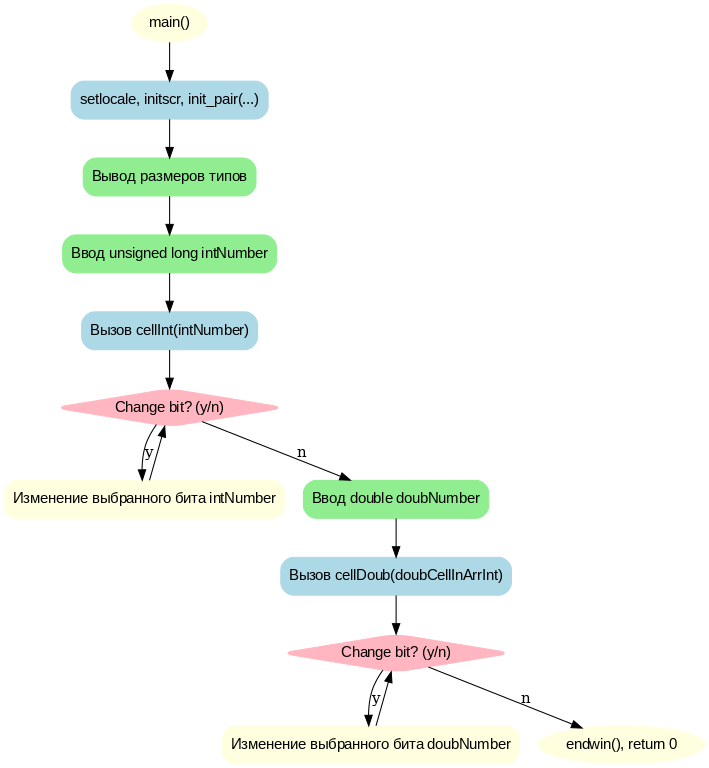
|  |  |  |
| --- | --- | --- |
| Студенты гр. 4373 |  | Шепелев Д.Н.  Дядюра Ю.С. |
| Преподаватель |  | Кочетков А.В. |

Санкт-Петербург

2024

****

**Блок-схемы алгоритмов**

Рис 1 Блок-схема функции main

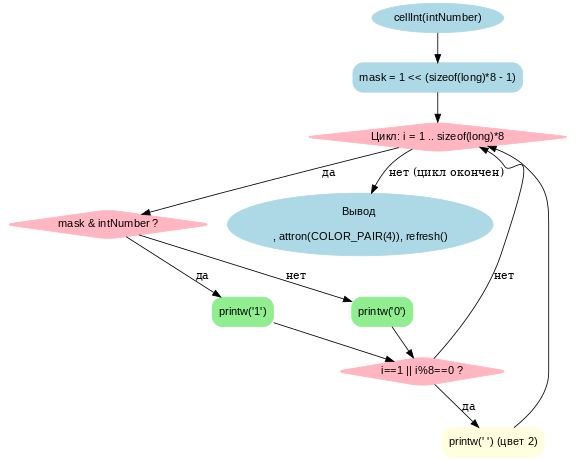


Рис 2 Блок-схема функции cellint

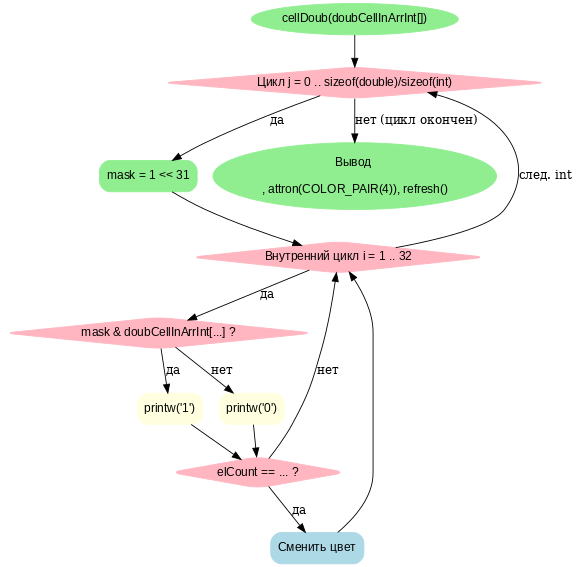


Рис 3 Блок-схема функции cellDoub

Код программы:

#include <iostream>

#include <ncurses.h>

using namespace std;

void cellInt(unsigned long intNumber) {

unsigned long mask = 1;

mask <<= sizeof(long) \* 8 - 1;

attron(COLOR\_PAIR(1));

for (long i = 1; i < sizeof(long) \* 8 + 1; ++i, mask >>= 1) {

mask & intNumber ? printw("1") : printw("0");

if (i == 1 || i % 8 == 0) {

attron(COLOR\_PAIR(2));

printw(" ");

}

}

printw( "\n\n" );

attron(COLOR\_PAIR(4));

refresh();

}

void cellDoub(int doubCellInArrInt[]) {

unsigned int mask, elCount = 0;

attron(COLOR\_PAIR(1));

for (int j = 0; j < sizeof(double) / sizeof(int); ++j) {

mask = 1;

mask <<= 31;

for (int i = 1; i < sizeof(int) \* 8 + 1; ++i, mask >>= 1) {

mask& doubCellInArrInt[sizeof(double) / sizeof(int) - 1 - j] ? printw("1") : printw("0");

elCount++;

if (elCount == (sizeof(double) / 8) \* 12) {

attron(COLOR\_PAIR(3));

printw(" ");

}

else if (elCount == 1) {

attron(COLOR\_PAIR(2));

printw(" ");

}

}

}

printw( "\n\n");

attron(COLOR\_PAIR(4));

refresh();

}

int main()

{

setlocale(LC\_ALL, "Russian");

initscr();

if (has\_colors() == FALSE){

endwin();

cout << "Your terminal does not support color\n)";

return 1;

}

start\_color();

init\_pair(1,COLOR\_RED, COLOR\_BLACK);

init\_pair(2,COLOR\_GREEN, COLOR\_BLACK);

init\_pair(3,COLOR\_BLUE, COLOR\_BLACK);

init\_pair(4,COLOR\_WHITE, COLOR\_BLACK);

int bitID, bitValue;

unsigned long intNumber;

unsigned long maskLong;

unsigned int maskBitID[2] = {0,0};

char input;

union {

float floatNumber;

int floatCellInInt;

};

union {

double doubNumber;

int doubCellInArrInt[sizeof(double) / sizeof(int)];

};

setlocale(0, "");

printw("int: %ld bytes\n", sizeof(int));

printw("short int: %ld bytes\n" , sizeof(short int));

printw("long int: %ld bytes\n", sizeof(long int));

printw("float: %ld bytes\n" , sizeof(float));

printw("double: %ld bytes\n" , sizeof(double));

printw("long double: %ld bytes\n" , sizeof(long double));

printw("char : %ld bytes\n", sizeof(char));

printw("bool: %ld bytes\n", sizeof(bool));

printw("\nEnter variable unsigned long: ");

refresh();

scanw("%ld",&intNumber);

printw("\n");

cellInt(intNumber);

do{

printw( "Change bit? (y/n): ");

refresh();

scanw("%c",&input);

if (input != 'n'){

printw( "\nBit place (right to left): ");

refresh();

scanw("%i",&bitID);

printw( "\nBit value: ");

refresh();

scanw("%i",&bitValue);

maskLong = 1;

maskLong <<= bitID;

if (bitValue == 1) {

if ((intNumber ^ maskLong) > intNumber) {

intNumber ^= maskLong;

}

}

else {

if ((intNumber ^ maskLong) < intNumber) {

intNumber ^= maskLong;

}

}

printw("\n");

cellInt(intNumber);

printw("Result number: %ld \n\n", intNumber);

refresh();

}

} while (input != 'n');

printw( "\nEnter variable double: ");

refresh();

scanw("%lf",&doubNumber);

printw("\n");

cellDoub(doubCellInArrInt);

do {

printw("Change bit? (y/n): ");

refresh();

scanw("%c",&input);

if (input != 'n') {

printw( "\nnBit place (right to left): ");

refresh();

scanw("%i",&bitID);

printw( "\nBit value: ");

refresh();

scanw("%i",&bitValue);

if (bitID < 32) {

maskBitID[0] = 0;

maskBitID[1] = 1;

maskBitID[1] <<= bitID;

}

else {

maskBitID[0] = 1;

maskBitID[1] = 0;

maskBitID[0] <<= bitID - 32;

}

if (bitValue == 1) {

if (((doubCellInArrInt[1] ^ maskBitID[0]) + (doubCellInArrInt[0] ^ maskBitID[1])) > doubCellInArrInt[0]+ doubCellInArrInt[1]) {

doubCellInArrInt[1] ^= maskBitID[0];

doubCellInArrInt[0] ^= maskBitID[1];

}

}

else {

if (((doubCellInArrInt[1] ^ maskBitID[0]) + (doubCellInArrInt[0] ^ maskBitID[1])) < doubCellInArrInt[0] + doubCellInArrInt[1]) {

doubCellInArrInt[1] ^= maskBitID[0];

doubCellInArrInt[0] ^= maskBitID[1];

}

}

printw("\n");

cellDoub(doubCellInArrInt);

printw("\nResult number: %f \n\n", doubNumber);

refresh();

}

} while (input != 'n');

endwin();

return 0;

}

Примеры запуска:

