МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №7**

по дисциплине: Основы программирования

тема: «Побитовые операции»

Выполнил: ст. группы ПВ-223

Пахомов Владислав Андреевич

Проверили:

Притчин Иван Сергеевич

Черников Сергей Викторович

Код-ревьер: ст. группы ПВ-223

Голуцкий Георгий Юрьевич

Белгород 2022 г.

**Лабораторная работа № 7**

**Вариант №1 (10 mod 10 + 1 = 1)**

**Содержание отчёта:**

* Тема лабораторной работы
* Цель лабораторной работы
* Решения задач
  + Условие задачи
  + Тестовые данные
  + Исходный код функции и её спецификация
* Работа над ошибками (код-ревью)
* Вывод по работе.

**Тема лабораторной работы:** Побитовые операции

**Цель лабораторной работы:** получение навыков работы с побитовыми операциями.

**Решения задач:**

1. Задача №1

Вывести восьмеричное представление записи числа 𝑥.

|  |  |  |
| --- | --- | --- |
|  | Входные данные | Выходные данные |
| 1 | 0 | 0 |
| 2 | 1437 | 2635 |
| 3 | 2739128 | 12345670 |
| 4 | 18446744073709551615 | 1777777777777777777777 |

1. Заголовок: void printOct(unsigned long long x)
2. Назначение: выводит восьмеричное представление числа x

|  |  |  |
| --- | --- | --- |
|  | Входные данные | Выходные данные |
| 1 | x = 010 = 08 = 02, at = 63 | 0 |
| 2 | x = 310 = 38 = 112, at = 0 | 3 |
| 3 | x = 7126310 = 213 1378 = 1’0001’0110’0101’11112, at = 57 | 1 |
| 4 | x = 922337203685477580810 = 1’000’000’000’000’000’000’0008 = 1000’0000’0000’0000’0000 0000’0000’0000’0000’0000’0000 0000’0000’0000’0000’00002, at = 0 | 1 |

1. Заголовок: char getOctDigit(unsigned long long x, int at)
2. Назначение: возвращает цифру в восьмеричном виде числа x на позиции at с левого конца числа в двоичном коде.

#include <stdio.h>

#include "1func.h"  
  
#define **OCT\_BLOCK\_BINARY\_SIZE** 3  
#define **OCT\_FULL\_BLOCK** 7ll  
#define **LONG\_LONG\_BIT\_SIZE** (sizeof(unsigned long long) << 3)  
  
*/\**

*\* Заголовок: char getOctDigit(unsigned long long x, int at)*

*\* Назначение: возвращает цифру в восьмеричном виде числа x на позиции at с левого*

*\* конца числа в двоичном коде.   
 \*/*char getOctDigit(unsigned long long x, int at) {  
 return (x >> (**LONG\_LONG\_BIT\_SIZE** - 1 - at)) & **OCT\_FULL\_BLOCK**;  
}  
  
*/\**

*\* Заголовок: void printOct(unsigned long long x)   
 \* Назначение: выводит восьмеричное представление числа x   
 \*/*void printOct(unsigned long long x) {  
 if (x == 0) {  
 printf("0\n");  
 return;  
 }  
  
 int digitBit = 0;  
 char digit;  
  
 while (!(digit = getOctDigit(x, digitBit)))  
 digitBit += **OCT\_BLOCK\_BINARY\_SIZE**;  
  
 while (digitBit < **LONG\_LONG\_BIT\_SIZE**) {  
 char currentDigit = getOctDigit(x, digitBit);  
 digitBit += **OCT\_BLOCK\_BINARY\_SIZE**;  
  
 printf("%d", currentDigit);  
 }  
  
 printf("\n");  
}

Ссылка на репозиторий: <https://github.com/IAmProgrammist/programming-and-algorithmization-basics/blob/c/lab7/sharedfuncs/src/1func.c>

1. Задача №4

Напишите функцию 𝑖𝑛𝑣𝑒𝑟𝑡𝐻𝑒𝑥, которая преобразует число 𝑥, переставляя в обратном порядке цифры в шестнадцатеричном представлении данного натурального числа.

|  |  |  |
| --- | --- | --- |
|  | Входные данные | Выходные данные |
| 1 | 010 = 02 = 016 | 010 = 02 = 016 |
| 2 | 6210 = 11’11102 = 3E16 | 22710 = 1110’00112 = E316 |
| 3 | 5117210 = 1100’0111’1110’01002 = C7E416 | 2009210 = 100’1110’0111’11002 = 4E7C16 |
| 4 | 131176846746379032010 = 0001’0010’0011’0100 0101’0110’0111’1000 1001’1010’1011’1100 1101’1110’1111’00002 = 123456789ABCDEF016 | 114779740903081654510 = 1111’1110’1101’1100 1011’1010’1001’1000 0111’0110’0101’0100 0011’0010’00012 = FEDCBA98765432116 |
| 5 | 1844674407370955161510 = 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’11112 = FFFFFFFFFFFFFFFF16 | 1844674407370955161510 = 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’11112 = FFFFFFFFFFFFFFFF16 |

1. Заголовок: void invertHex(unsigned long long \*x)
2. Назначение: преобразует значение по адресу x переставляя в обратном порядке цифры значения по адресу x в обратном порядке в шестнадцатеричном представлении.

#include "4func.h"  
  
#define **HEX\_FULL\_BLOCK** 15  
#define **HEX\_BLOCK\_LEN** 4  
  
/\*

\* Заголовок: void invertHex(unsigned long long \*x)  
 \* Назначение: преобразует значение по адресу x переставляя в обратном порядке цифры

\* значения по адресу x в обратном порядке в шестнадцатеричном

\* представлении.   
 \*/  
void invertHex(unsigned long long \*x) {  
 unsigned long long reversedX = 0;  
  
 while (\*x) {  
 reversedX = (reversedX << **HEX\_BLOCK\_LEN**) + (\*x & **HEX\_FULL\_BLOCK**);  
 \*x >>= **HEX\_BLOCK\_LEN**;  
 }  
  
 \*x = reversedX;  
}

Ссылка на репозиторий: <https://github.com/IAmProgrammist/programming-and-algorithmization-basics/blob/c/lab7/sharedfuncs/src/4func.c>

1. Задача №7

Определить максимальную длину последовательности подряд идущих битов, равных единице в двоичном представлении данного целого числа.

|  |  |  |
| --- | --- | --- |
|  | Входные данные | Выходные данные |
| 1 | x = 010 = 02 | 0 |
| 2 | x = 110 = 12 | 1 |
| 3 | x = 51110 = 1’1111’11112 | 9 |
| 4 | x = 63310 = 10’0111’1001 | 3 |
| 5 | x = 1844674407370955161510 = 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’11112 | 64 |

1. Заголовок: int getMaximumLength(unsigned long long x)
2. Назначение: возвращает максимальное количество подряд идущих единиц в двоичной записи числа x.

#include "7func.h"  
  
/\*

\* Заголовок: int getMaximumLength(unsigned long long x)  
 \* Назначение: возвращает максимальное количество подряд идущих единиц в двоичной

\* записи числа x.   
 \*/  
int getMaximumLength(unsigned long long x) {  
 int maxLen = 0;  
 int currLen = 0;  
  
 while (x) {  
 if (x & 1) {  
 currLen += 1;  
  
 if (currLen > maxLen)  
 maxLen = currLen;  
 } else  
 currLen = 0;  
  
 x >>= 1;  
 }  
  
 return maxLen;  
}

Ссылка на репозиторий: <https://github.com/IAmProgrammist/programming-and-algorithmization-basics/blob/c/lab7/sharedfuncs/src/7func.c>

1. Задача №8

\*\* Выполнить циклический сдвиг в двоичном представлении данного натурального числа 𝑥 на 𝑘 битов влево.

|  |  |  |
| --- | --- | --- |
|  | Входные данные | Выходные данные |
| 1 | x = 010 = 02  k = 100 | 010 = 02 |
| 2 | x = 110 = 12  k = 100 | 110 = 12 |
| 3 | x = 3410 = 10’00102  k = 0 | 3410 = 10’00102 |
| 4 | x = 3410 = 10’00102  k = 1 | 510 = 1012 |
| 5 | x = 25310 = 1111’11012  k = 5 | 19110 = 1011’11112 |
| 5 | x = 2310 = 1’01112  k = 3 | 1510 = 11112 |

1. Заголовок: void cycleShift(unsigned long long \*pX, int k)
2. Назначение: выполняет циклический сдвиг числа по адресу pX на k влево.

#include "8func.h"  
  
/\*

\* Заголовок: void cycleShift(unsigned long long \*pX, int k)   
 \* Назначение: выполняет циклический сдвиг числа по адресу pX на k влево.   
 \*/  
void cycleShift(unsigned long long \*pX, int k) {  
 if (\*pX == 0)  
 return;  
  
 while (k--) {  
 unsigned long long copyX = \*pX;  
 \*pX = 0;  
 int pow = 1;  
  
 while (copyX & (~1)) {  
 \*pX += (copyX & 1) << pow;  
 copyX >>= 1;  
 pow++;  
 }  
  
 \*pX += 1;  
 }  
}

Ссылка на репозиторий: <https://github.com/IAmProgrammist/programming-and-algorithmization-basics/blob/c/lab7/sharedfuncs/src/8func.c>

P.S. по тестовым данным пособия невозможно определить точно механизм работы циклического сдвига произвольного размера. Так, например, переносится ли ноль при циклическом сдвиге? (прим. x = 101112, k = 3; cycleShift(x, k) -> x = 11112 или cycleShift(x, k) -> x = 111012?).

1. Задача №9

\*\* Дано длинное целое неотрицательное число. Получить число, удалив каждую вторую цифру в двоичной записи данного числа, начиная со старших цифр.

|  |  |  |
| --- | --- | --- |
|  | Входные данные | Выходные данные |
| 1 | x = 25310 = 1111’11012 | 1410 = 11102 |
| 2 | x = 46710 = 1’1101’00112 | 2910 = 1’11012 |

1. Заголовок: unsigned long long removeEverySecondDigit(unsigned long long x)
2. Назначение: возвращает преобразованное число x, в котором удаляется каждая вторая цифра в двоичной записи числа, начиная со старших цифр.

#include "9func.h"  
  
/\*

\* Заголовок: unsigned long long removeEverySecondDigit(unsigned long long x)  
 \* Назначение: возвращает преобразованное число x, в котором удаляется каждая вторая

\* цифра в двоичной записи числа, начиная со старших цифр.   
 \*/  
unsigned long long removeEverySecondDigit(unsigned long long x) {  
 unsigned long long variantA = 0, variantB = 0;  
  
 int i;  
 for (i = 0; x; x >>= 1, i++) {  
 unsigned long long digit = x & 1;  
  
 if (i & 1)  
 variantA += digit << (i >> 1);  
 else  
 variantB += digit << (i >> 1);  
 }  
  
 if (i & 1)  
 return variantB;  
 else  
 return variantA;  
}

Ссылка на репозиторий: <https://github.com/IAmProgrammist/programming-and-algorithmization-basics/blob/c/lab7/sharedfuncs/src/9func.c>

1. Задача №10

\*\* Дано целое неотрицательное число. Получить число перестановкой битов

каждого байта данного числа в обратном порядке.

|  |  |  |
| --- | --- | --- |
|  | Входные данные | Выходные данные |
| 1 | x = 010 = 02 | 0 |
| 2 | x = 5568110 = 1101’1001’1000’00012 | 3324110 = 1000’0001’1101’10012 |
| 3 | x = 1407864610 = 1101’0110’1101’0010’1011’01102 | 1198152610 = 1011’0110 1101’0010’1101’01102 |
| 4 | x = 1844674407370955161510 = 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’11112 | 1844674407370955161510 = 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’1111 1111’1111’1111’11112 |

1. Заголовок: unsigned long long getInvertedByte(unsigned long long x)
2. Назначение: возвращает преобразованное значение x переставляя в обратном порядке байты значения x в обратном порядке

#include "10func.h"  
  
#define **BYTE\_FULL\_BLOCK** 255  
#define **BYTE\_BLOCK\_LEN** 8  
  
/\*

\* Заголовок: unsigned long long getInvertedByte(unsigned long long x)   
 \* Назначение: возвращает преобразованное значение x переставляя в обратном порядке

\* байты значения x в обратном порядке

\*/  
unsigned long long getInvertedByte(unsigned long long x) {  
 unsigned long long reversedByteNumber = 0;  
  
 while (x) {  
 reversedByteNumber = (reversedByteNumber << **BYTE\_BLOCK\_LEN**) +

(x & **BYTE\_FULL\_BLOCK**);  
 x >>= **BYTE\_BLOCK\_LEN**;  
 }  
  
 return reversedByteNumber;  
}

Ссылка на репозиторий: <https://github.com/IAmProgrammist/programming-and-algorithmization-basics/blob/c/lab7/sharedfuncs/src/10func.c>

1. Пакеты с монетами (1037A)

#include <stdio.h>  
  
int main() {  
 int cash;  
 scanf("%d", &cash);  
  
 int cashBags = 0;  
  
 while (cash != 0) {  
 *// Ну надо же хотя бы один побитовый оператор вставить  
 // По побитовым операциям же лаба ¯\\_(ツ)\_/¯* cash >>= 1;  
 cashBags++;  
 }  
  
 printf("%d", cashBags);  
   
 return 0;  
}



Ссылка на репозиторий: <https://github.com/IAmProgrammist/programming-and-algorithmization-basics/blob/c/lab7/>1037A.c

**Код-ревью:**

Упростить первую задачу

**Работа над ошибками (код-ревью)**

Упростил

**Вывод:** в ходе выполнения лабораторной работы получены навыки работы с побитовыми операциями.