МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное бюджетное образовательное учреждение высшего образования

«Севастопольский государственный университет»

Кафедра «Информатика и вычислительная техника»

**ОТЧЁТ**

по лабораторной работе №3

по теме: «Обработка целочисленных  
данных»

по дисциплине: «Программирование»

Выполнил:

стд. гр. ИВТ/б-19-2-о

Садриев А.Э.

Проверил:

Оболенский Д. М.

Севастополь

2019

1. **Цель работы**

Закрепить навыки объявления переменных и констант; освоить операции  
для данных целых типов; закрепить навыки применения оператора  
присваивания.

2. **Постановка задачи**

1) Ознакомиться с принципами хранения и обработки целочисленных  
данных в Java.  
2) Разработать и отладить программу, демонстрирующую выполнение  
операций над данными целого типа.

**3.** **Вариант №23**

|  |  |  |  |
| --- | --- | --- | --- |
| Номер варианта | A | b | c |
| 23 | 456 | 1234 | (2\*a+b/45)\*3%14 |

**4. Текст программы**

public class Test1

{

public static void main( String [] arg)

{

int a = 456;

int b = 1234;

int c;

System.out.printf("a =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(a),a,a,a);

System.out.printf("b =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(b),b,b,b);

c = -a;

System.out.printf("-a =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(c),c,c,c);

c = a+b;

System.out.printf("a+b =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(c),c,c,c);

c = a-b;

System.out.printf("a-b =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(c),c,c,c);

c = a\*b;

System.out.printf("a\*b =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(c),c,c,c);

c = a/b;

System.out.printf("a/b =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(c),c,c,c);

c = a%b;

System.out.printf("a%%b =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(c),c,c,c);

a++;

System.out.printf("a++ =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(a),a,a,a);

b--;

System.out.printf("b-- =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(b),b,b,b);

a--;

b++;

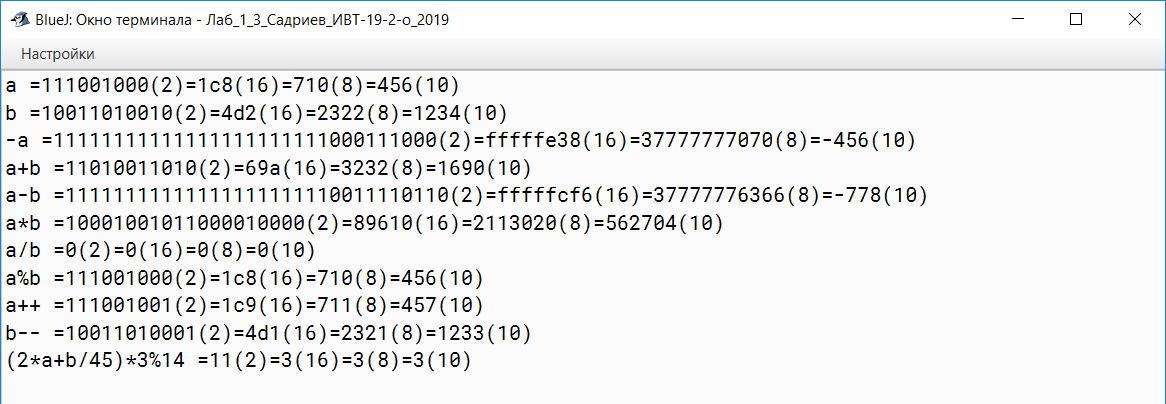
c = (2\*a+b/45)\*3%14;

System.out.printf("(2\*a+b/45)\*3%%14 =%s(2)=%h(16)=%o(8)=%d(10)\n",Integer.toBinaryString(c),c,c,c);

}

}

**5. Результат программы**

****

**6. Вывод**

В ходе лабораторной работы была разработана программа, выполняющая некоторые арифметические действия и выводящая результат на экран.



**Внутреннее и внешнее представление целочисленных данных**

Рисунок 8.1

Рисунок 8.2

Внутренне представление – это то, которое «видит» процессор, внешнее – то, которое видит и применяет при записи литералов (в данном случае, целых чисел) программист. Внутреннее представление целочисленных данных – двоичное. Под знак выделен левый бит. Значение 0 этого бита имеют положительные числа, значение 1 – отрицательные. Различные целочисленные типы отличаются длиной (числом битов n). Для типа byte n=8, для типа short n=16, для типа int n=32, для типа long n=64. Чем больше n, тем шире диапозон представления чисел.

Отрицательные числа представляются Java в дополнительных кодах следующим образом:

Знак=1;

Значение\* = 2n-1 – |Значение|

*где* Значение\* *- значение числа в дополнительном коде,*

Значение *– обычное представление двоичного числа.*

Принято, что для любого положительного числа дополнительный код совпадает с прямым кодом (обычным представлением числа в двоичной позиционной системе счисления).

Все операции с целыми числами выполняются в двоичной системе счисления. Использование дополнительного кода для представления двоичных чисел позволяет процессору корректно выполнять операции сложения и вычитания на двоичном сумматоре.

Литералы (целые числа, присутствующие в программе) автоматически переводятся из внешнего представления (последовательность символов) во внутреннее (последовательность битов двоичного целого числа). Метод println() автоматически преобразует числа из внутреннего во внешнее представление.

**Рисование схем в документе**

При составлении технического отчета используется различный графический материал, например схемы алгоритмов вычислительных процессов, схемы обмена данными, структурные схемы вычислительных систем. Подобная информация может быть представлена как на отдельных чертежах, так и в виде рисунков в основном тексте отчета. Правила оформления рисунков рассмотрены в **п. 2.8.1**. Для построения схем можно использовать возможности офисного приложения MicrosoftWord или специализированной системы для рисования подобных объектов, например, Visio. Перечисленные приложения снабжены примитивами для рисования различных блоков, а также графическими элементами: прямая, эллипс, дуга, кривая **и т. п.** С основами работы в Microsoft Visio можно ознакомиться в книге **Бермана Н. Д.** «MS Visio 2010: основы работы».

При построении схем алгоритмов используются 4 основных вида блоков: «начало», «конец», «операционный блок» («процесс») и «проверка условия» («решение»). В качестве дополнительных применяют блок «подготовка», «начало цикла», «конец цикла», «ссылка на текущую страницу», «ссылка на другую страницу», «ручная операция», «данные» и другие [4]. Примеры схемы алгоритма и структурной схемы приведены ниже на рисунках 8.3 и 8.4.

y=0

Рисунок 8.3 – Фрагмент схемы алгоритма

Формирование ꝣi

xi=a+(b-a)\*ꝣi

y=y+xi

Рисунок 8.4 – Структурная схема

\Принята

Принята

Отброшена

Нет ошибки

Ошибка 1 рода

Ошибка 1 рода

Отброшена

Неверна

℔0:

Верна

i

l n

Использование

y



**Установка ОС. Программного**

**Обеспечения. Дайверов.**

**Антивирусных программ.**

**+7 999 199-66-55**

**+7 977 951-35-46**

**Выезд на дом в удобное для Вас время**

**АЛАН**