Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)

Факультет «Робототехника и комплексная автоматизация»

Кафедра «Системы автоматизированного проектирования»

**Отчет по лабораторной работе**

По курсу «Объектно-ориентированное программирование»

Выполнил: Студент Сергеева Д.К.

Группа РК6-26Б

Проверил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2020 г.

**ПЯВУ Вариант 20В**

**Задание:**

Разработать объектно-ориентированную программу для перечисления всех латинских букв, которые одновременно отсутствуют или совпадают в двух любых подмножествах, заданных двумя символьными строками аргументов команды её вызова. Искомый результат должна отображать строка стандартного вывода, где все буквы перечисляются в обратном алфавитном порядке. Программная реализация таких перечислений должна быть основана на разработке класса множества латинских букв с компонентными методами перегрузки операторов ‘^’ и ‘~’ для выполнения операций исключения совпадений и дополнения. Они должны быть ориентированы на обработку поля приватных данных, где состав множества фиксируется двоичными разрядами целого числа. Конструктор класса должен обеспечивать преобразование исходной символьной записи множества в двоичный целочисленный формат. Необходимо также предусмотреть перегрузку оператора класса потока стандартного вывода для отображения объектов класса множества латинских букв.

**Алгоритм:**

В нашем класс хранится число, кодирующее двоичные разряды символьной строки, компонентные методы перегрузки операторов “^” и “~”, и дружественную функцию перегрузки “ <<”. Для перевода строки в двоичное число сначала узнаем порядковый номер буквы в алфавите, сдвигаем на это количество битов единицу, и побитовой операцией ИЛИ записываем 1 на определенном месте в нашем числе, предварительно проверив, что это буква латинского алфавита, а не другой символ. Для выполнения оператора исключения совпадений перегружаем оператор “^”. Для выполнения оператора дополнения перегружаем оператор “~”. Применив операцию исключения совпадений к 2 множествам, находим все латинские буквы, которые встречаются либо в 1 строке, либо в другой. Потом применив операцию дополнения, находим все латинские буквы, которые отсутствуют в этих строках или одновременно присутствуют.

**Входные данные:**

2 символьные строки, состоящие из латинских букв.

**Выходные данные:**

1 символьная строка, состоящая из латинских букв.

**Текст программы:**

#include <iostream>

#include <cctype>

using namespace std;

class Alpha {

private:

unsigned int bin;

public:

Alpha() { bin = 0; };

Alpha(char\*);

Alpha operator ^(Alpha&);

Alpha operator ~();

friend ostream& operator << (ostream&, Alpha&);

};

Alpha::Alpha(char\* s)

{

bin = 0;

while (\*s)

{

if (((\*s >= 'A') && (\*s <= 'Z')) || ((\*s >= 'a') && (\*s <= 'z')))

{

bin |= (1 << (tolower(\*s) - 'a'));

}

s++;

}

}

Alpha Alpha::operator ^ (Alpha& a)

{

Alpha z;

z.bin = this->bin ^ a.bin;

return(z);

}

Alpha Alpha::operator ~ ()

{

Alpha z;

z.bin = (~(this->bin));

return(z);

}

ostream& operator << (ostream& out, Alpha& a)

{

unsigned bit = 33554432;

for (int i = 25; i >= 0; i--)

{

if ((a.bin & bit) > 0)

out << (char)('a' + i);

bit = bit >> 1;

}

return out;

}

int main(int argc, char\* argv[])

{

if (argc != 3)

{

cout << "Invalid number of arguments" << endl;

return -1;

}

Alpha x(argv[1]);

Alpha y(argv[2]);

Alpha z;

z = x ^ y;

z = ~z;

cout << z << endl;

return 0;

}

**Тесты:**

|  |  |  |
| --- | --- | --- |
| **Номер** | **Вход** | **Выход** |
| 1 | AbcKlnop Bcdefghi | zyxwvutsrqmjcb |
| 2 | abc | Invalid number of arguments |

**Список использованной литературы:**

* Волосатова Т.М., Родионов С.В. Лекции по курсу «Объектно-ориентированное программирование»
* bigor.bmstu.ru