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 Петраков С.А.

Группа РК6-26Б

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

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2020 г.

**Вариант ПЯВУ В16**

**Задание:**

Разработать объектно-ориентированную программу для перечисления всех латинских букв, которые отсутствуют в двух любых символьных строках, заданных аргументами 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 компонентным методом перегрузки операторов ‘~’ и ‘+’ для выполнения теоретико-множественных операций дополнения и объединения. 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.

Следует также предусмотреть перегрузку оператора класса потока стандартного вывода для отображения объектов класса множества латинских букв.

**Алгоритм:**

Для реализации данной программы я создал класс, в котором будет 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. Для выполнения оператора объединения перегружаем оператор “+”. Для выполнения оператора дополнения перегружаем оператор “~”. Применив операцию объединения к 2 множествам, находим все латинские буквы, которые присутствуют в обоих строках. Потом применив операцию дополнения, находим все латинские буквы, которые отсутствуют в этих строках.

**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();

Alpha(char\*);

Alpha operator+(Alpha&);

Alpha operator~();

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

};

Alpha::Alpha()

{

bin = 0;

}

Alpha::Alpha(char\* str)

{

bin = 0;

while (\*str != '\0')

{

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

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

str++;

}

}

Alpha Alpha::operator+(Alpha& other)

{

Alpha result;

result.bin = this->bin | other.bin;

return result;

}

Alpha Alpha::operator~()

{

Alpha result;

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

return result;

}

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

{

unsigned bit = 33554432;

int i;

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

{

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

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

bit = bit >> 1;

}

return out;

}

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

{

if (argc != 3)

{

std::cout << "Not count arguments.\n";

return -1;

}

Alpha first(argv[1]);

Alpha second(argv[2]);

Alpha summa = first + second;

Alpha result = ~summa;

cout << result << endl;

return 0;

}

**Тесты:**

|  |  |  |
| --- | --- | --- |
| **Номер** | **Вход** | **Выход** |
| 1 | AbcFgh CdeZijp | yxwvutsqonmlk |
| 2 | abc | Not count arguments |

**Список использованной литературы:**

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