

2. (1)

3. (2)

4. (3)

5. (4)

6. Используем арифметические характеристические функции

*U={1,2,3,4,5,6,7}*

*A = {1, 3, 5, 7}, B = {2, 3, 6, 7}, C = {4, 5, 6, 7}*

*F2 =*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *0* | *1* | *0* | *1* | *0* | *1* |
|  | *0* | *1* | *1* | *0* | *0* | *1* | *1* |
|  | *1* | *1* | *1* | *0* | *1* | *1* | *1* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *0* | *0* | *1* | *1* | *1* | *1* |
|  | *1* | *1* | *1* | *0* | *1* | *1* | *1* |
|  | *0* | *0* | *0* | *1* | *0* | *0* | *0* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *X* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *1* | *1* | *0* | *0* | *1* | *1* |
|  | *0* | *0* | *0* | *1* | *1* | *1* | *1* |
|  | *0* | *1* | *1* | *0* | *0* | *0* | *0* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *X* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *1* | *1* | *0* | *0* | *0* | *0* |
|  | *0* | *0* | *0* | *1* | *0* | *0* | *0* |
|  | *0* | *1* | *1* | *1* | *0* | *0* | *0* |

*АХФ для F2 = 5610 = 1110002*

*F3 =*

*D = ; E =*

*F3 =*

*Вычислим D*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *X* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *1* | *1* | *0* | *0* | *1* | *1* |
|  | *0* | *0* | *0* | *1* | *1* | *1* | *1* |
|  | *0* | *1* | *1* | *0* | *0* | *0* | *0* |

*Вычислим E*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *0* | *1* | *0* | *1* | *0* | *1* |
|  | *0* | *1* | *1* | *0* | *0* | *1* | *1* |
|  | *1* | *1* | *0* | *0* | *1* | *1* | *0* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *0* | *1* | *0* | *1* | *0* | *1* |
|  | *0* | *1* | *1* | *0* | *0* | *1* | *1* |
|  | *0* | *0* | *1* | *0* | *0* | *0* | *1* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *0* | *0* | *1* | *1* | *0* |
|  | *0* | *0* | *1* | *0* | *0* | *0* | *1* |
|  | *1* | *1* | *1* | *0* | *1* | *1* | *1* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *0* | *0* | *1* | *1* | *1* | *1* |
|  | *1* | *1* | *1* | *0* | *1* | *1* | *1* |
|  | *0* | *0* | *0* | *1* | *0* | *0* | *0* |

*Вычислим F3=*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *1* | *1* | *0* | *0* | *0* | *0* |
|  | *0* | *0* | *0* | *1* | *0* | *0* | *0* |
|  | *0* | *1* | *1* | *1* | *0* | *0* | *0* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *1* | *1* | *0* | *0* | *0* | *0* |
|  | *0* | *0* | *0* | *1* | *0* | *0* | *0* |
|  | *0* | *0* | *0* | *0* | *0* | *0* | *0* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *1* | *1* | *1* | *0* | *0* | *0* |
|  | *0* | *0* | *0* | *0* | *0* | *0* | *0* |
|  | *0* | *1* | *1* | *1* | *0* | *0* | *0* |

*АХФ для F3 = 5610 = 1110002*

*Так как АХФ для F3 и F2 совпали, F3 и F2 тождественны.*

*7. Для вычисления результатов выражений используем класс CBool :*

class CBool {  
public:  
 bool data;  
  
 CBool(bool data) {  
 this->data = data;  
 }  
  
 CBool operator^(CBool aSet) {  
 return data ^ aSet.data;  
 }  
  
 CBool operator\*(CBool aSet) {  
 return CBool(data && aSet.data);  
 }  
  
 CBool operator\*(int num) {  
 return CBool(this->data && num);  
 }  
  
 CBool operator+(CBool aSet) {  
 return CBool(data || aSet.data);  
 }  
  
 CBool operator!() {  
 return CBool(!data);  
 }  
  
 CBool operator-(CBool aSet) {  
 return \*this \* !aSet;  
 }  
};

*Где:*

*^ – симметрическая разность*

*\* - пересечение*

*+ - объединение*

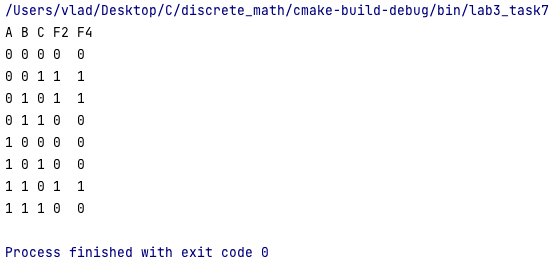
*! – дополнение*

*- - разница*

Итоговая программа выглядит следующим образом

int main() {  
 std::cout << "A B C F2 F4" << std::endl;  
 for (int bA = 0; bA <= 1; bA++)  
 for (int bB = 0; bB <= 1; bB++)  
 for (int bC = 0; bC <= 1; bC++) {  
 CBool A = CBool(bA);  
 CBool B = CBool(bB);  
 CBool C = CBool(bC);  
 CBool F2 = (B - C) + (C - (A + B));  
 CBool F4 = ((B - C) ^ (C - ((A ^ B) ^ (A - (A - B))))) ^ (  
 (B - C) - ((B - C) - (C - ((A ^ B) ^ (A - (A - B))))));  
 std::cout << bA << " " << bB << " " << bC << " " << F2.data << " " << F4.data << std::endl;  
 }  
   
 return 0;  
}

Результат выполнения:



Таблицы истинности совпали, следовательно F2 и F4 тождественны.

8. Используя функции, полученные в л.р. № 1.1 получим класс Sett:

class Sett {  
public:  
 std::vector<int> elements;  
  
 Sett(std::vector<int> elms) {  
 elements = elms;  
 std::sort(elements.begin(), elements.end());  
 }  
  
 ~Sett() {  
 }  
  
 Sett operator\*(Sett anotherSet) {  
 std::vector<int> arrayC(0, 0);  
 int arrayASize = elements.size();  
 int arrayBSize = anotherSet.elements.size();  
 size\_t i = 0, j = 0;  
  
 while (i < arrayASize && j < arrayBSize)  
 if (elements[i] < anotherSet.elements[j])  
 i++;  
 else if (elements[i] > anotherSet.elements[j])  
 j++;  
 else {  
 arrayC.push\_back(elements[i]);  
 i++;  
 j++;  
 }  
  
 return Sett(arrayC);  
 }  
  
 Sett operator-(Sett anotherSet) {  
 std::vector<int> arrayC(0, 0);  
 int arrayASize = elements.size();  
 int arrayBSize = anotherSet.elements.size();  
 size\_t i = 0, j = 0;  
  
 while (i < arrayASize && j < arrayBSize)  
 if (elements[i] < anotherSet.elements[j])  
 arrayC.push\_back(elements[i++]);  
 else if (elements[i] > anotherSet.elements[j])  
 j++;  
 else {  
 i++;  
 j++;  
 }  
  
 while (i < arrayASize)  
 arrayC.push\_back(elements[i++]);  
  
 return Sett(arrayC);  
 }  
  
 Sett operator+(Sett anotherSet) {  
 std::vector<int> arrayC(0, 0);  
 int arrayASize = elements.size();  
 int arrayBSize = anotherSet.elements.size();  
 size\_t i = 0, j = 0;  
  
 while (i < arrayASize && j < arrayBSize)  
 if (elements[i] < anotherSet.elements[j])  
 arrayC.push\_back(elements[i++]);  
 else if (elements[i] > anotherSet.elements[j])  
 arrayC.push\_back(anotherSet.elements[j++]);  
 else {  
 arrayC.push\_back(elements[i]);  
 i++;  
 j++;  
 }  
  
 while (i < arrayASize)  
 arrayC.push\_back(elements[i++]);  
  
 while (j < arrayBSize)  
 arrayC.push\_back(anotherSet.elements[j++]);  
  
 return Sett(arrayC);  
 }  
  
 Sett non(Sett universum) {  
 std::vector<int> arrayC(0, 0);  
 int arraySize = elements.size();  
 int universumSize = universum.elements.size();  
 size\_t i = 0, j = 0;  
 *// Проверяем, что универсум действительно универсум* **assert**(elements[arraySize - 1] <= universum.elements[universumSize - 1]);  
  
 while (i < universumSize && j < arraySize) {  
 if (universum.elements[i] < elements[j])  
 arrayC.push\_back(universum.elements[i++]);  
 else if (universum.elements[i] == elements[j]) {  
 i++;  
 j++;  
 *// вторым его отличием будет то, что если элемент есть в A и его нет в universum, программа будет падать* } else  
 **assert**(elements[j] >= universum.elements[i]);  
 }  
  
 while (i < universumSize)  
 arrayC.push\_back(universum.elements[i++]);  
  
 return Sett(arrayC);  
 }  
  
 Sett operator^(Sett anotherSet) {  
 std::vector<int> arrayC(0, 0);  
 int arrayASize = elements.size();  
 int arrayBSize = anotherSet.elements.size();  
 size\_t i = 0, j = 0;  
  
 while (i < arrayASize && j < arrayBSize)  
 if (elements[i] < anotherSet.elements[j])  
 arrayC.push\_back(elements[i++]);  
 else if (elements[i] > anotherSet.elements[j])  
 arrayC.push\_back(anotherSet.elements[j++]);  
 else {  
 j++;  
 i++;  
 }  
  
 while (i < arrayASize)  
 arrayC.push\_back(elements[i++]);  
  
 while (j < arrayBSize)  
 arrayC.push\_back(anotherSet.elements[j++]);  
  
 return Sett(arrayC);  
 }  
  
 void print() {  
 for (int i = 0; i < elements.size(); i++) {  
 std::cout << elements[i] << " ";  
 }  
  
 std::cout << std::endl;  
 }  
};

*Где:*

*^ – симметрическая разность*

*\* - пересечение*

*+ - объединение*

*non – дополнение*

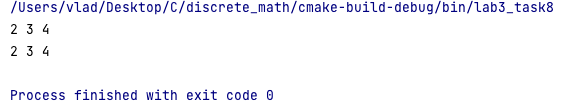
*- - разница*

*print – метод, выводящий множество*

*Итоговая программа выглядит так:*

int main() {  
 Sett A(std::vector<int>(**{**1, 3, 5, 7**}**));  
 Sett B(std::vector<int>(**{**2, 3, 6, 7**}**));  
 Sett C(std::vector<int>(**{**4, 5, 6, 7**}**));  
 Sett U(std::vector<int>(**{**1, 2, 3, 4, 5, 6, 7**}**));  
  
 Sett F3 = ((B - C) ^ (C - ((A ^ B) ^ (A \* B)))) ^ ((B - C) \* (C - ((A ^ B) ^ (A \* B))));  
 Sett F4 = ((B - C) ^ (C - ((A ^ B) ^ (A - (A - B))))) ^ ((B - C) - ((B - C) - (C - ((A ^ B) ^ (A - (A - B))))));  
  
 F3.print();  
 F4.print();  
  
 return 0;  
}

Результат выполнения программы:



Результаты вычислений совпали, следовательно множества F3 и F4 тождественны.

2. (1)

3. (2)

4. (3)

5. (4)

6. Используем арифметические характеристические функции

*U={2,3,4,5,6,7}*

*A={2,3,5,6}, B={2,5,6,7}, C={4,5,6,7}*

*F2 =*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *0* | *1* | *1* | *0* |
|  | *1* | *0* | *0* | *1* | *1* | *1* |
|  | *1* | *1* | *0* | *1* | *1* | *1* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *0* | *1* | *1* | *1* | *1* |
|  | *1* | *1* | *0* | *1* | *1* | *1* |
|  | *0* | *0* | *1* | *0* | *0* | *0* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *X* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *0* | *1* | *1* | *0* |
|  | *0* | *0* | *1* | *1* | *1* | *1* |
|  | *1* | *1* | *0* | *0* | *0* | *0* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *X* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *0* | *0* | *0* | *0* |
|  | *0* | *0* | *1* | *0* | *0* | *0* |
|  | *1* | *1* | *1* | *0* | *0* | *0* |

*АХФ для F2 = 2810 = 111002*

*F3 =*

*D = ; E =*

*F3 =*

*Вычислим D*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *X* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *0* | *1* | *1* | *0* |
|  | *0* | *0* | *1* | *1* | *1* | *1* |
|  | *1* | *1* | *0* | *0* | *0* | *0* |

*Вычислим E*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *0* | *1* | *1* | *0* |
|  | *1* | *0* | *0* | *1* | *1* | *1* |
|  | *0* | *1* | *0* | *0* | *0* | *1* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *0* | *1* | *1* | *0* |
|  | *1* | *0* | *0* | *1* | *1* | *1* |
|  | *1* | *0* | *0* | *1* | *1* | *0* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *1* | *0* | *0* | *0* | *1* |
|  | *1* | *0* | *0* | *1* | *1* | *0* |
|  | *1* | *1* | *0* | *1* | *1* | *1* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *0* | *0* | *1* | *1* | *1* | *1* |
|  | *1* | *1* | *0* | *1* | *1* | *1* |
|  | *0* | *0* | *1* | *0* | *0* | *0* |

*Вычислим F3=*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *0* | *0* | *0* | *0* |
|  | *0* | *0* | *1* | *0* | *0* | *0* |
|  | *1* | *1* | *1* | *0* | *0* | *0* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *0* | *0* | *0* | *0* |
|  | *0* | *0* | *1* | *0* | *0* | *0* |
|  | *0* | *0* | *0* | *0* | *0* | *0* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x* | *2* | *3* | *4* | *5* | *6* | *7* |
|  | *1* | *1* | *1* | *0* | *0* | *0* |
|  | *0* | *0* | *0* | *0* | *0* | *0* |
|  | *1* | *1* | *1* | *0* | *0* | *0* |

*АХФ для F3 = 2810 = 111002*

*Так как АХФ для F3 и F2 совпали, F3 и F2 тождественны.*

*7. Для вычисления результатов выражений используем код из л.р. 1.1 boolsetoperations.*

*U={0,1,2,3,4,5}*

*A={0,1,3,4}, B={0,3,4,5}, C={2,3,4,5}*

*F2 =*

*Функция для вычисления ЛХФ*

*…*

*F4 =*

*E =*

*D =*

*F4 =*