Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №1**

По дисциплине «Теоретические и множественные основы интеллектуальных систем»

Тема: «Множества. Работа с битовой маской.»

**Выполнил:**

Студент 1 курса

Группы ИИ-21

Карагодин Д.Л.

**Проверил:**

Глущенко Т.А.

Брест 2022

**Цель работы:** Реализовать в программе предлагаемый метод случайного поиска.

**Вариант 4**

**Ход работы:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4. | {1,2,3,4,5} | {4,5,6,7} | {3,5,7,8} |  |

1. **Для заданного множества A построить булеан.** **Реализовать на выбор сортировку слиянием или объединение слиянием. При сортировке слиянием на вход подается неупорядоченное множество (неотсортированный массив). Выход – упорядоченное множество (отсортированный по возрастанию массив). При объдинении сливаем два упорядоченных массива.** **Реализовать все операции над множествами:объединение, пересечение, разность, симметрическую разность, дополнение, используя битовую маску.** **Вычислить программно , используя битовую маску, выражение согласно варианту, проиллюстрировать результат диаграммой Эйлера-Венна.** **Указать для номера вашего варианта (порядок генерации) бинарный код Грея.**

**Ход работы:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <cmath>

#include <iostream>

#include <string>

#include <malloc.h>

using namespace std;

int u[] = { 1,2,3,4,5,6,7,8,9,10,11 };

int codeGrey(int n) {

return n ^ (n >> 1);}

void bulean(int d[],int razmer) {

char\* maska = new char[razmer];

char\* buff\_maska = new char[razmer+1];

short count = 0;

for (int i = 0; i < pow(2, razmer);i++,count++)

{

\_itoa(count, buff\_maska, 2);

for (int k = 0; k<= strlen(buff\_maska); k++) {

maska[razmer - strlen(buff\_maska)+k] = buff\_maska[k];

}

cout << "{";

for (int j =0; j<razmer; j++) if (maska[razmer-1-j] == '1') cout << d[j]<<" ";

cout << "}"<<endl;

}}

bool\* bit\_maska(int\* a) {

bool\* bit\_mask = (bool\*)calloc(11, sizeof(short));

for (int i = 0; i < 11; i++) bit\_mask[i] = 0;

for (int i = 0; i < 11; i++) {

for (int j = 0; j < 11; j++) {

try { if (a[i] == u[j]) { bit\_mask[j] = 1; }

}

catch (exception& e) { i == 11; break; }

}

}

return bit\_mask;}

int\* obed(int\* a, int\* b) {

bool\* bit\_a = bit\_maska(a);

bool\* bit\_b = bit\_maska(b);

int\* result = (int\*) calloc(1,sizeof(int));

result[0] = 0;

short count = 1;

for (int i = 0; i < 11; i++) {

if (bit\_a[i] == 1 || bit\_b[i] == 1) {

result = (int\*)realloc(result, sizeof(int) \* count);

result[count-1] = u[i];

cout << result[count-1] << " ";

count++;

}

}

cout << endl;

return result;}

int\* pereceh(int a[], int b[]) {

bool\* bit\_a = bit\_maska(a);

bool\* bit\_b = bit\_maska(b);

int\* result = (int\*) calloc(1,sizeof(int));

short count = 1;

for (int i = 0; i < 11; i++) {

if (bit\_a[i] == 1 && bit\_b[i] == 1) {

result = (int\*)realloc(result, sizeof(int) \* count);

result[count - 1] = u[i];

cout << result[count - 1] << " ";

count++;

}

} cout << endl;

return result;}

int\* raznoct(int a[], int b[]) {

bool\* bit\_a = bit\_maska(a);

bool\* bit\_b = bit\_maska(b);

int\* result = (int\*) calloc(1,sizeof(int));

short count = 1;

for (int i = 0; i < 11; i++) {

if (bit\_a[i] == 1 && bit\_b[i] != 1) {

result = (int\*)realloc(result, sizeof(int) \* count);

result[count - 1] = u[i];

cout << result[count - 1] << " ";

count++;

}

} cout << endl;

return result;}

int\* simmrazn(int a[], int b[]) {

bool\* bit\_a = bit\_maska(a);

bool\* bit\_b = bit\_maska(b);

int\* result = (int\*) calloc(1,sizeof(int));

short count = 1;

for (int i = 0; i < 11; i++) {

if (bit\_a[i] != 1 != bit\_b[i] != 1) {

result = (int\*)realloc(result, sizeof(int) \* count);

result[count - 1] = u[i];

cout << result[count - 1] << " ";

count++;

}

} cout << endl;

return result;}

int\* dopoln(int a[]) {

bool\* bit\_a = bit\_maska(a);

int\* result = (int\*) calloc(1,sizeof(int));

short count = 1;

for (int i = 0; i < 11; i++) {

if (bit\_a[i] != 1) {

result = (int\*)realloc(result, sizeof(int) \* count);

result[count - 1] = u[i];

cout << result[count - 1] << " ";

count++;

}

} cout << endl;

return result;}

int main()

{

int a[] = { 1,2,3,4,5 };

int b[] = { 4,5,6,7 };

int c[] = { 3,5,7,8 };

bulean(a, 5);

int d[9] = { 0 };

int i = 0, j = 0, k = 0;

/\* while (i != 5 && j != 4)

{

int data;

if (a[i] < b[j])

{d[k] = a[i];i++;k++;}

else

if (a[i] > b[j])

{d[k] = b[j];j++;k++;}

else

{d[k] = a[i];i++;j++;k++;}}

if (i == 5 && j == 4);

else

{

if (i != 5){

while (i != 5)

{d[k] = a[i];k++;i++;}

}

if (j != 4){

while (j != 4)

{d[k] = b[j];k++;j++;}}}

for (int m = 0; m < k; m++) cout << d[m] << " ";\*/

return 0;

}

Вывод: Я умный.