**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ.УЛЬЯНОВА (ЛЕНИНА)**

Кафедра вычислительной техники

Отчет по лабораторной работе №1 на тему:

«Множества»

по дисциплине «Алгоритмы и структуры данных»

Костичев С.В.

Санкт-Петербург

2018

Шалашников М.И.

Преподаватель

Студент гр. 7307

Оглавление

[Цель работы 3](#_Toc526114370)

[Задание 3](#_Toc526114371)

[Временная сложность 3](#_Toc526114372)

[Формализация 3](#_Toc526114373)

[Контрольные примеры 4](#_Toc526114374)

[Измерения времени 5](#_Toc526114375)

[Вывод 6](#_Toc526114376)

[Код программы 7](#_Toc526114377)

# **Цель работы**

Сравнительные испытания способов хранения множеств в памяти ЭВМ.

# **Задание**

Вычислить множество, содержащее цифры, имеющиеся в любом из множеств А, B, C, D.

# **Временная сложность**

|  |  |  |
| --- | --- | --- |
| Тип представления данных | Ожидаемая временная сложность. | Фактическая временная сложность. |
| Массив элементов | O(n^2) | O(n^2) |
| Список | O(n^2) | O(n^2) |
| Массив битов | O(1) | O(1) |
| Машинное слово | O(1) | O(1) |

# **Формализация**

E = A | B | C | D

# **Контрольные примеры**

|  |  |
| --- | --- |
| Имя множества | Элементы множества |
| A | { } |
| B | { } |
| C | { } |
| D | { } |
| Res\_1 (результат) | { } |

|  |  |
| --- | --- |
| Имя множества | Элементы множества |
| A | { } |
| B | { 1, 2, A } |
| C | { 4, 5, 6 } |
| D | { 6, B, F } |
| Res\_1 (результат) | { 1, 2, 4, 5, 6, A, B, F} |

|  |  |
| --- | --- |
| Имя множества | Элементы множества |
| A | { 3, 6, 9 } |
| B | { 1, 2, 3, 5, 7, C} |
| C | { A, B, C, D, E, F } |
| D | {0,2, 4, 6, 8,} |
| Res\_1 (результат) | {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F} |

# **Измерения времени**

|  |  |  |  |
| --- | --- | --- | --- |
|  | N = 300 000 | N = 500 000 | N = 1 000 000 |
| Массив | 158 | 247 | 553 |
| Список | 2131 | 3439 | 6959 |
| Массив битов | 16 | 24 | 56 |
| Машинное слово | 1 | 1 | 2 |

\*N – кол-во измерений

# Вывод

В ходе данной работы были получены практические знания по обработке множеств. Так как память под массив выделяется до начала работы с ним, могут возникнуть проблемы с пустыми множествами и не до конца заполненными множествами (память выделена, а элементов нет). Эта проблема решается, если для представления множества в памяти используется односвязный список. Память в таком случае выделяется под каждый элемент множества отдельно, и ее ровно столько, сколько необходимо. Но существует недостаток – необходимо хранить с каждым элементом множества указатель на следующий элемент и тратить время на работу с ним. Если мощность универсума (m) не очень велика и существует функция отображения элемента множества в соответствующий ему порядковый номер бита, то гораздо компактнее и быстрее будет использовать вектор битов. Временная мощность такого способа O(m), но т.к. m – фиксированная, то получаем O(1). Вектор битов может быть представлен более компактно в форме машинного слова. Если мощность универсума превосходит разрядность переменных, то можно использовать два слова. Главный недостаток такого способа – отсутствие удобного доступа к каждому биту машинного слова.

Первые два способа более просты для понимания и требуют меньших знаний, но в то же время требует больше кода. Третий и четвертый способ быстрее, компактнее, но сложнее для понимания, требуют знаний о представлении данных в памяти пк (машинное слово), необходимо придумать функцию отображения (массив битов). Так же у каждого способа есть свои плюсы и минусы, о которых нужно помнить во время разработки.

# **Код программы**

#include <time.h>

#include <iostream>

#include <cstring>

using namespace std;

enum { max\_arr = 17, max\_arr\_res = 68, experiment = 300000};

void Randomize(char Arr[]) {

int wA = 0;

wA = rand() % 0xFFFF;

for (int i = 0, k = 0; i < max\_arr - 1; i++)

if ((wA >> i) & 1)

Arr[k++] = (char)(i <= 9 ? i + '0' : i + 'A' - 10);

}

struct mas {

char a;

mas \*next;

mas(char b = '\0', mas \*p = nullptr) {

next = p;

a = b;

}

~mas() {

delete next;

}

};

void out(mas \*p) {

while (p != nullptr) {

cout << p->a << " ";

p = p->next;

}

cout << endl;

}

bool check(mas \*p, char a) {

while (p != nullptr && p->a != a) {

p = p->next;

}

return (p) ? true : false;

}

int main() {

srand(time(nullptr));

char a[max\_arr] = {}, b[max\_arr] = {}, c[max\_arr] = {}, d[max\_arr] = {}, res\_1[max\_arr\_res] = {};

/\*std::cin.getline(a, 16,'\n');

std::cin.getline(b, 16,'\n');

std::cin.getline(c, 16,'\n');

std::cin.getline(d, 16,'\n');\*/

// - - - - - - - ARRAYS - - - - - - - - - -

Randomize(a);

Randomize(b);

Randomize(c);

Randomize(d);

cout << "Generated array A: ";

cout << a << endl;

cout << "Generated array B: ";

cout << b << endl;

cout << "Generated array C: ";

cout << c << endl;

cout << "Generated array D: ";

cout << d << endl;

int k = 0; // 1.1 array;

bool flag;

unsigned long t1 = clock();

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

for (int i = 0; a[i] != '\0'; i++) {

res\_1[i] = a[i];

k++;

}

for (int i = 0; b[i] != '\0'; i++) {

flag = false;

for (int i1 = 0; i1 < k && !flag; i1++) {

if (b[i] == res\_1[i1]) {

flag = true;

}

}

if (!flag) {

res\_1[k++] = b[i];

}

}

for (int i = 0; c[i] != '\0'; i++) {

flag = false;

for (int i1 = 0; i1 < k && !flag; i1++) {

if (c[i] == res\_1[i1]) {

flag = true;

}

}

if (!flag) {

res\_1[k++] = c[i];

}

}

for (int i = 0; d[i] != '\0'; i++) {

flag = false;

for (int i1 = 0; i1 < k && !flag; i1++) {

if (d[i] == res\_1[i1]) {

flag = true;

}

}

if (!flag) {

res\_1[k++] = d[i];

}

}

}

unsigned long t2 = clock();

unsigned long t = t2 - t1;

cout << "Result array: " << res\_1 << endl;

cout << "Array time = " << t << endl;

mas \*A = new mas(a[0]);

mas \*B = new mas(b[0]);

mas \*C = new mas(c[0]);

mas \*D = new mas(d[0]);

mas \*res = nullptr;

for (int i = 1; a[i] != '\0'; i++) {

A = new mas(a[i], A);

}

for (int i = 1; b[i] != '\0'; i++) {

B = new mas(b[i], B);

}

for (int i = 1; c[i] != '\0'; i++) {

C = new mas(c[i], C);

}

for (int i = 1; d[i] != '\0'; i++) {

D = new mas(d[i], D);

}

// - - - - - - - - LIST - - - - - - - - - - - -

mas \*delA = A;

mas \*delB = B;

mas \*delC = C;

mas \*delD = D;

cout << "Work with list" << endl;

t1 = clock();

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

A = delA;

B = delB;

C = delC;

D = delD;

res = nullptr;

for (A; A != nullptr; A = A->next) { //1.1 list;

res = new mas(A->a, res);

}

for (B; B != nullptr; B = B->next) {

if (!check(res, B->a)) {

res = new mas(B->a, res);

}

}

for (C; C != nullptr; C = C->next) {

if (!check(res, C->a)) {

res = new mas(C->a, res);

}

}

for (D; D != nullptr; D = D->next) {

if (!check(res, D->a)) {

res = new mas(D->a, res);

}

}

if(i != experiment - 1) delete res;

}

t2 = clock();

t = t2 - t1;

cout << "Result list: ";

out(res);

cout << "List time = " << t << endl;

// - - - - - - - - - - BOOL ARRAYS - - - - - - - - - -

cout << "Work with boll arrays" << endl;

bool vA[max\_arr - 1] = { false };

bool vB[max\_arr - 1] = { false };

bool vC[max\_arr - 1] = { false };

bool vD[max\_arr - 1] = { false };

bool vO[max\_arr - 1] = { false };

for (int i = 0; vA[i]; i++)

vA[a[i] <= '9' ? a[i] - '0' : a[i] - 'A' + 10] = true;

for (int i = 0; vB[i]; i++)

vB[b[i] <= '9' ? b[i] - '0' : b[i] - 'A' + 10] = true;

for (int i = 0; vC[i]; i++)

vC[c[i] <= '9' ? c[i] - '0' : c[i] - 'A' + 10] = true;

for (int i = 0; vD[i]; i++)

vD[d[i] <= '9' ? d[i] - '0' : d[i] - 'A' + 10] = true;

for (int i = 0; i < max\_arr - 1; i++) {

vO[i] = vA[i] || vB[i] || vC[i] || vD[i];

}

t1 = clock();

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

for (int i = 0; i < max\_arr - 1; i++) {

vO[i] = vA[i] || vB[i] || vC[i] || vD[i];

}

}

t2 = clock();

t = t2 - t1;

for (int i = 0, k = 0; i < max\_arr - 1; i++)

if (vO[i]) res\_1[k++] = (char)(i <= 9 ? i + '0' : i + 'A' - 10);

cout << "Result bool array: ";

for (int i = 0;res\_1[i] != '\0'; i++) {

cout << res\_1[i];

}

cout << endl << "Bool array time: " << t << endl;

//- - - - - - - - - - -MACHINE WORD - - - - - - - - -

cout << "Work with Machine word" << endl;

for (int i = 0; i < max\_arr - 1; i++) {

res\_1[i] = 0;

}

short wA = 0, wB = 0, wC = 0, wD = 0;

for (int i = 0; a[i]; i++)

wA |= (1 << (a[i] <= '9' ? a[i] - '0' : a[i] - 'A' + 10));

for (int i = 0; b[i]; i++)

wB |= (1 << (b[i] <= '9' ? b[i] - '0' : b[i] - 'A' + 10));

for (int i = 0; c[i]; i++)

wC |= (1 << (c[i] <= '9' ? c[i] - '0' : c[i] - 'A' + 10));

for (int i = 0; d[i]; i++)

wD |= (1 << (d[i] <= '9' ? d[i] - '0' : d[i] - 'A' + 10));

int wO = 0;

cout << "Result: " << endl << "wA: " << wA << "\n" << "wB: " << wB << "\n" << "wC: " << wC << "\n" << "wD: " << wD << "\n";

t1 = clock();

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

wO = wA | wB | wC | wD;

}

t2 = clock();

for (int i = 0, k = 0; i < max\_arr - 1; i++)

if ((wO >> i) & 1) res\_1[k++] = (char)(i <= 9 ? i + '0' : i + 'A' - 10);

t = t2 - t1;

cout << "Result Machine word: ";

for (int i = 0;res\_1[i] != '\0'; i++) {

cout << res\_1[i];

}

cout << endl << "Machine word time: " << t << endl;

delete res;

delete delA;

delete delB;

delete delC;

delete delD;

std::cin.get();

return 0;

}