Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«Белгородский государственный технологический университет им. В.Г. Шухова»

(БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №13

По дисциплине: «Основы программирования»

**Тема**: «Множества»

Выполнил: студент группы ВТ-231

Борченко Александр Сергеевич

Проверили:

Черников Сергей Викторович

Новожен Никита Викторович

**Цель работы:** закрепление навыков работы со структурами, изучение простых способов представления множеств в памяти ЭВМ.

# Содержание работы:

| <b>Задание №1.</b> Выполнить реализацию множества на типе uint32_t. | 2  |
|---|----|
| Содержимое файла <i>bitset.h</i>                                    | 3  |
| Задание №2. На неупорядоченном массиве                              | 6  |
| Задание №3. * На упорядоченном массиве                              | 14 |
| Задачи с code forces  |    |
| Задание №1. Определи маршрут (1056А)                                | 20 |
| Задание №2. Пропущенная серия (440А)                                | 22 |
| <b>Задание №3.</b> Перестановка букв (1093В)                        | 23 |
| <b>Задание №4.</b> Тихий класс (1166A)                              | 25 |
| Задание №5. Щедрый Кефа (841А)                                      | 26 |
| <b>Задание №6.</b> Перекраска собачек (1025A)                       | 27 |
| Задание №7. Ступени (1011А)   | 28 |
| Задание №8. Башни (37А)   | 30 |
| Задание №9. Бейджик (1020В)   | 32 |
| Задание №10. Разнообразие – это хорошо (672В)                       | 33 |
| Задание №11. Игра: Банковские карты (777В)                          | 34 |
| Задание №12. Противоположности притягиваются (131В)                 | 36 |

# **Задание №1.** Выполнить реализацию множества на типе uint32\_t. Содержимое файла *bitset.h*.

```
#ifndef БИБЛИОТЕКИ BIT SET
#define БИБЛИОТЕКИ BIT SET H
#include <stdio.h>
#include <assert.h>
#include <malloc.h>
    bitset set;
    set.maxValue = maxValue;
    uint32 t arraySize = (maxValue + 31) / 32;
    return (set.values[index] & (1 << bitOffset)) != 0;</pre>
    uint32 t arraySize = (set1.maxValue + 31) / 32;
    for (uint32 t i = 0; i < arraySize; ++i) {
```

```
ool bitset isSubset(bitset set1, bitset set2) {
    if (set1.maxValue > set2.maxValue) {
    uint32_t arraySize = (set1.maxValue + 31) / 32;
    set->values = NULL;
    bitset result = bitset create(set1.maxValue > set2.maxValue ?
            bitset insert(&result, i);
bitset bitset intersection(bitset set1, bitset set2) {
    bitset result = bitset create(set1.maxValue < set2.maxValue ?</pre>
set1.maxValue : set2.maxValue);
    bitset result = bitset create(set1.maxValue);
```

## Задание №2. На неупорядоченном массиве

```
#ifndef C UNORDERED SET H
#define C UNORDERED SET H
#include <stdio.h>
#include <assert.h>
#include <malloc.h>
unordered_array_set unordered_array_set_create(size_t capacity) {
    set.data = (int *)malloc(capacity * sizeof(int));
    set.capacity = capacity;
            if (subset.data[i] == set.data[j]) {
    return (*(int*)a - *(int*)b);
```

```
if (set1.size != set2.size) {
qsort(set1.data, set1.size, sizeof(int), compare);
qsort(set2.data, set2.size, sizeof(int), compare);
    size t new capacity = (set->capacity == 0) ? 1 : set->capacity * 2;
    int *new data = realloc(set->data, new capacity * sizeof(int));
   set->data = new data;
set->data[set->size++] = value;
```

```
unordered array set insert(&intersection, set1.data[i]);
unordered array set set2) {
   for (size t i = 0; i < set1.size; i++) {
        if (unordered array set in(set2, set1.data[i]) == set1.size) {
   return difference;
   unordered array set complement = {NULL, 0, 0};
       if (unordered array set in(set, universumSet.data[i]) == set.size) {
   return complement;
   free(set.data);
unordered_array_set_symmetricDifference(unordered array set set1,
   unordered_array_set diff2 = unordered_array_set_difference(set2, set1);
```

```
printf("%d ", set.data[i]);
}
printf("}\n");
}
#endif //C_UNORDERED_SET_H
```

#### Тесты:

```
#include "unordered set.h"
    unordered array set set1 = unordered array set create from array((int[])
    unordered array set set2 = unordered array set create from array((int[])
    unordered array set expectedSet =
unordered array set create from array((int[]) {1, 2, 3}, 3);
    assert (unordered array set isEqual(resSet, expectedSet));
    unordered array set delete(expectedSet);
    unordered array set set1 = unordered array set create from array((int[])
    unordered array set set2 = unordered array set create from array((int[])
    unordered_array_set expectedSet =
unordered_array_set_create_from_array((int[]) {1, 2, 3}, 3);
    assert(unordered_array_set_isEqual(resSet, expectedSet));
    unordered_array_set_delete(set1);
    unordered_array_set_delete(set2);
    unordered_array_set_delete(resSet);
    unordered array set delete(expectedSet);
    unordered_array_set set = {NULL, 0, 0};
    size t position = unordered array set in(set, 10);
    assert(position == 1);
    unordered array set delete(set);
    unordered array set set = {NULL, 0, 0};
    unordered array set insert(&set, 5);
    unordered array set insert(&set, 10);
    size t position = unordered array set in(set, 15);
    assert(position == set.size);
    unordered array set delete(set);
    test unordered array set in1();
```

```
unordered_array_set emptySet = {NULL, 0, 0};
    assert(!unordered array set isSubset(emptySet, set));
    test unordered array set isSubset1();
    test unordered array set isSubset2();
    unordered_array_set_insert(&set, 5);
unordered_array_set_insert(&set, 10);
unordered_array_set_deleteElement(&set, 5);
    unordered_array_set_insert(&set1, 10);
    unordered_array_set_insert(&set2, 10);
set2);
```

```
unordered array set delete(intersection);
   unordered_array_set_insert(&set2, 10);
   unordered array set difference = unordered array set difference (set1,
   unordered array set delete(set1);
   unordered array set delete(set2);
   unordered array set delete(difference);
   unordered array set set2 = {NULL, 0, 0};
   unordered array set insert(&set1, 5);
unordered array set symmetricDifference(set1, set2);
   unordered array set delete(set2);
   unordered array set delete(symmetric difference);
   unordered array set complement = unordered array set complement(set,
universumSet);
   assert(complement.size == 1 && complement.data[0] == 15);
   unordered array set delete(complement);
```

```
test_unordered_array_set_union();
  test_unordered_array_set_deleteElement();
  test_unordered_array_set_intersection();
  test_unordered_array_set_difference();
  test_unordered_array_set_symmetricDifference();
  test_unordered_array_set_complement();
}

int main() {
  test();
  return 0;
}
```

# Задание №3. \* На упорядоченном массиве

```
#ifndef БИБЛИОТЕКИ ORDERED SET H
#define БИБЛИОТЕКИ ORDERED SET H
#include <stdint.h>
#include <assert.h>
#include <malloc.h>
ordered array set ordered array set create(size t capacity) {
    set.capacity = capacity;
ordered array set ordered array set create from array(const int *a, size t
        set.data[i] = a[i];
    return set;
size_t ordered_array_set_in(ordered_array_set *set, int value) {
    for (size t i = 0; i < set1.size; ++i) {</pre>
       if (set1.data[i] != set2.data[i]) {
bool ordered array set isSubset(ordered array set subset, ordered array set
set) {
    for (size t i = 0; i < subset.size; ++i) {</pre>
```

```
set->data[i] = value;
    ++set->size;
        set->size--;
ordered array set ordered array set union (ordered array set set1,
ordered array set set2) {
    ordered_array_set result = ordered array set create(set1.size +
set2.size);
        } else if (set2.data[j] < set1.data[i]) {</pre>
ordered array set set2) {
```

```
} else {
             result.data[k++] = set1.data[i++];
ordered array set ordered array set difference (ordered array set setl,
        if (set1.data[i] < set2.data[j]) {</pre>
    while (i < set1.size) {</pre>
set1, ordered array set set2) {
    ordered array set result1 = ordered array set difference(set1, set2);
    free(result2.data);
set);
    free(set.data);
```

#### Тесты:

```
#include <assert.h>
   ordered array set set = ordered array set create from array((int[]) {1,
    ordered array set set1 = ordered array set create from array((int[]) {1,
   ordered array set set2 = ordered array set create from array((int[]) {1,
   ordered array set set3 = ordered array set create from array((int[]) {1,
    assert(ordered_array_set_isSubset(set2, set1) == false);
assert(ordered_array_set_isSubset(set1, set3) == false);
    ordered_array_set_delete(set1);
    ordered array set delete(set3);
   ordered_array_set_insert(&set, 3);
   ordered array set set1 = ordered array set create from array((int[]) {1,
   ordered array set set2 = ordered array set create from array((int[]) {2,
    ordered array set expectedSet =
ordered array set create from array((int[]) {1, 2, 3, 4, 5}, 5);
   assert(ordered array set isEqual(result, expectedSet) == true);
   ordered array set delete(set2);
   ordered array set delete (result);
    ordered array set delete(expectedSet);
   ordered array set set = ordered array set create from array((int[]) {1,
    ordered array set expectedSet =
ordered array set create from array((int[]) {1, 2, 4, 5}, 4);
    assert(ordered array set isEqual(set, expectedSet) == true);
```

```
ordered array set delete(expectedSet);
    ordered array set set1 = ordered array set create from array((int[]) {1,
   ordered array set set2 = ordered array set create from array((int[]) {2,
    assert(ordered_array_set_isEqual(result, expectedSet) == true);
    ordered array set delete(expectedSet);
   ordered array set set1 = ordered array set create from array((int[]) {1,
    ordered array set expectedSet =
ordered array set create from array((int[]) {1, 4}, 2);
    assert(ordered array set isEqual(result, expectedSet) == true);
   ordered array set delete(set2);
    ordered array set delete(result);
    ordered array set delete(expectedSet);
    ordered array set set1 = ordered array set create from array((int[]) {1,
    ordered array set set2 = ordered array set create from array((int[]) {2,
    ordered array set result = ordered array set symmetricDifference(set1,
    ordered array set expectedSet =
    assert(ordered array set isEqual(result, expectedSet) == true);
    ordered_array_set_delete(set1);
    ordered array set delete(expectedSet);
   ordered array set set = ordered array set create from array((int[]) {1,
ordered array set create from array((int[]) {1, 2, 3, 4, 5, 6, 7, 8, 9, 10},
10);
universumSet);
    ordered_array_set expectedSet =
ordered_array_set_create_from_array((int[]) {6, 7, 8, 9, 10}, 5);
   assert(ordered_array_set_isEqual(result, expectedSet) == true);
```

```
ordered_array_set_delete(expectedSet);
}

void test() {
   test_ordered_array_set_in();
   test_ordered_array_set_isSubset();
   test_ordered_array_set_insert();
   test_ordered_array_set_union();
   test_ordered_array_set_deleteElement();
   test_ordered_array_set_intersection();
   test_ordered_array_set_difference();
   test_ordered_array_set_symmetricDifference();
   test_ordered_array_set_complement();
}

int main() {
   test();
   return 0;
}
```

#### Задание №1. Определи маршрут (1056А)

```
#include <stdio.h>
size t linearSearch(const int a[], const size t n, int x) {
    return (unordered array set) {malloc(sizeof(int) * capacity), 0,
size t unordered array set in(unordered array set* set, int value) {
        unordered array set isAbleAppend(set);
         append(set -> data, &set -> size, value);
    unordered array set set = unordered array set create(new capacity);
         if (unordered_array_set_in(&set2, set1.data[i]) != set2.size)
    unordered_array_set_insert(&set, set1.data[i]);
    return set;
```

```
int main() {
    int n;
    scanf("%d", &n);
    int first_set_r;
    scanf("%d", &first_set_r);

unordered_array_set set = unordered_array_set_create(first_set_r);
    for (int I = 0; i < first_set_r; i++) {
        int x;
        scanf("%d", &x);

        unordered_array_set_insert(&set, x);
}

for (int i = 0; i < n - 1; i++) {
        scanf("%d", &first_set_r);

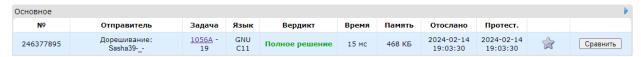
        unordered_array_set subset = unordered_array_set_create(first_set_r);
        for (int j = 0; j < first_set_r; j++) {
            int x;
            scanf("%d", &x);

            unordered_array_set_insert(&subset, x);
        }

        set = unordered_array_set_intersection(set, subset);
}

for (int i = 0; i < set.size; i++) {
    printf("%d ", set.data[i]);
    }

    return 0;
}</pre>
```



# Задание №2. Пропущенная серия (440А)

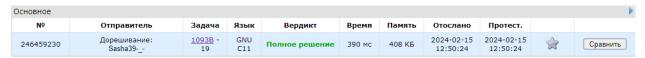
#### Код задачи:

```
#include <stdio.h>
unordered array set unordered array set create(size t capacity) {
   return (unordered array set) {malloc(sizeof(int) * capacity), 0,
capacity};
ordered array set ordered array set create(size t capacity) {
    return (ordered array set) {malloc(sizeof(int) * capacity), 0, capacity};
       set.data[i - 1] = i;
```

| Основное  |                         |                 |            |                |       |        |                        |                        | <b>▶</b> |
|-----------|-------------------------|-----------------|------------|----------------|-------|--------|------------------------|------------------------|----------|
| Nº        | Отправитель             | Задача          | Язык       | Вердикт        | Время | Память | Отослано               | Протест.               |          |
| 246452769 | Дорешивание:<br>Sasha39 | <u>440A</u> - 9 | GNU<br>C11 | Полное решение | 31 MC | 656 KB | 2024-02-15<br>11:45:26 | 2024-02-15<br>11:45:26 | Сравнить |

# Задание №3. Перестановка букв (1093В)

```
#include <stdio.h>
unordered array set unordered array set create(size t capacity) {
    return (unordered array set) {malloc(sizeof(int) * capacity), 0,
capacity};
        int is palindrome = 1;
        for (size t i = 0; i < eng alphabet; i++) {
            if (amount.data[i] != 0 && amount.data[i] != size str)
                is palindrome = 0;
```



#### Задание №4. Тихий класс (1166А)

#### Код задачи:

```
#include <stdio.h>
#include <stdlib.h>
unordered array set unordered array set create(size t capacity) {
   return (unordered array set) {malloc(sizeof(int) * capacity), 0,
capacity};
   unordered array set amount = unordered array set create(eng alphabet);
        int amount first class = amount.data[i] / 2;
        res += count couple(amount first class) +
count couple(amount second class);
```

| Основное  |                         |                |            |                |       |        |                        |                        | Þ        |
|-----------|-------------------------|----------------|------------|----------------|-------|--------|------------------------|------------------------|----------|
| Nº        | Отправитель             | Задача         | Язык       | Вердикт        | Время | Память | Отослано               | Протест.               |          |
| 246468412 | Дорешивание:<br>Sasha39 | <u>1166A</u> - | GNU<br>C11 | Полное решение | 15 MC | 260 KB | 2024-02-15<br>14:27:06 | 2024-02-15<br>14:27:06 | Сравнить |

# Задание №5. Щедрый Кефа (841А)

#### Код задачи:

```
#include <stdio.h>
int main() {
   int ammount_balls;
   int ammount_friends;
   scanf("%d %d", %ammount_balls, %ammount_friends);

   char balls[ammount_balls];
   scanf("%s", balls);

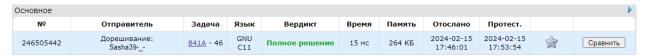
   int count_balls[123] = {0};//от 0 до 122 - количество букв в таблице

ASCII
   for(int i = 0; i < ammount_balls; i++) {
      count_balls[balls[i]]++;
   }

   int flag = 1;
   for(int i = 'a'; i <= 'z'; i++) {//Переводя в таблицу ASCII - от 97 до

122 включительно
      if(count_balls[i] > ammount_friends) {
        flag = 0;
      }
   }
   if(flag) {
        printf("YES\n");
   } else {
        printf("No\n");
   }

   return 0;
}
```



# Задание №6. Перекраска собачек (1025А)

#### Код задачи:

```
#include <stdio.h>
} ordered_array set;
ordered array set ordered array set create(size t capacity) {
   set.data = (int *) malloc(capacity * sizeof(int));
   set.capacity = capacity;
            flag = 1;
```

| Основное  |                         |                      |            |                |       |        |                        |                        | <b>▶</b> |
|-----------|-------------------------|----------------------|------------|----------------|-------|--------|------------------------|------------------------|----------|
| Nō        | Отправитель             | Задача               | Язык       | Вердикт        | Время | Память | Отослано               | Протест.               |          |
| 246557841 | Дорешивание:<br>Sasha39 | <u>1025A</u> -<br>23 | GNU<br>C11 | Полное решение | 31 мс | 344 КБ | 2024-02-15<br>19:20:23 | 2024-02-15<br>19:20:23 | Сравнить |

# Задание №7. Ступени (1011А)

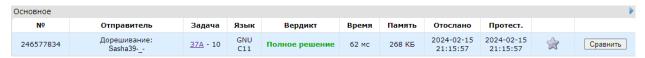
```
#include <stdio.h>
unordered array set unordered array set create(size t capacity) {
   return (unordered array set) {malloc(sizeof(int) * capacity), 0,
capacity};
    int amount stages = 1;
    for (int i = pos + 2; amount stages < k && i < amount letter; i++) {
        if (a[i] != 0 \&\& i - last > 1) {
    if (amount stages != k)
   char str[ammount available teps];
    for (size t i = 0; i < ammount available teps; i++) {</pre>
```

```
int result = get_minimum_total_weight_rocket(amount.data,
index_first_element, ammount_required_steps);
printf("%d", result);
return 0;
}
```



#### **Задание №8.** Башни (37A)

```
#include <stdio.h>
#include <malloc.h>
#include <stdlib.h>
\#define length limit 1001//по условию не превосходит 1000 (но включая 1000)
unordered array set unordered array set create(size t capacity) {
    return (unordered array set) {malloc(sizeof(int) * capacity), 0,
capacity};
        amount.data[length bars]++;
    int max tower height ammount = getMax(amount.data, amount.size);
```



# Задание №9. Бейджик (1020В)

#### Код задачи:

```
#include <stdio.h>
unordered_array_set;
unordered array set unordered array set create(size t capacity) {
   return (unordered array set) {malloc(sizeof(int) * capacity), 0,
capacity};
   int current = pos;
    int result = a[current] + 1;
    free(visited);
   int ammount pupils;
    int pupils[ammount pupils];
    for (size t i = 0; i < ammount pupils; i++) {</pre>
    for (int pos = 0; pos < ammount pupils; pos++) {</pre>
        printf("%d ", getReference(pupils, ammount pupils, pos));;
```

| Основное  |                         |                      |            |                |       |        |                        |                        | <b>▶</b> |
|-----------|-------------------------|----------------------|------------|----------------|-------|--------|------------------------|------------------------|----------|
| Nο        | Отправитель             | Задача               | Язык       | Вердикт        | Время | Память | Отослано               | Протест.               |          |
| 246581263 | Дорешивание:<br>Sasha39 | <u>1020B</u> -<br>16 | GNU<br>C11 | Полное решение | 31 MC | 260 KB | 2024-02-15<br>21:45:47 | 2024-02-15<br>21:45:47 | Сравнить |

# Задание №10. Разнообразие – это хорошо (672В)

#### Код задачи:

```
#include <stdio.h>
unordered array set unordered array set create(size t capacity) {
    int line length;
    scanf("%d", &line length);
   char str[line length];
    if (line length > eng alphabet)
```

| Основное  |                         |                  |            |                |       |        |                        |                        | •        |
|-----------|-------------------------|------------------|------------|----------------|-------|--------|------------------------|------------------------|----------|
| Nº        | Отправитель             | Задача           | Язык       | Вердикт        | Время | Память | Отослано               | Протест.               |          |
| 246582698 | Дорешивание:<br>Sasha39 | <u>672B</u> - 41 | GNU<br>C11 | Полное решение | 31 MC | 344 KB | 2024-02-15<br>21:59:37 | 2024-02-15<br>21:59:37 | Сравнить |

## Задание №11. Игра: Банковские карты (777В)

```
#include <stdlib.h>
#include <stdio.h>
unordered array set unordered array set create(size t capacity) {
   set.data = (int *)malloc(capacity * sizeof(int));
   set.size = capacity;
   set.capacity = capacity;
        set.data[i] = a[i] - '0';
    free (set.data);
    char Moriarty[ammount digits];
unordered array set create from array(Moriarty, ammount digits);
```

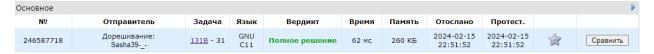


## Задание №12. Противоположности притягиваются (131В)

#### Код задачи:

```
#include <stdio.h>
#include <stdlib.h>
   set.data = (long long *)malloc(capacity * sizeof(long long));
   set.size = capacity;
       int client parameter;
       scanf("%d", &client parameter);
       array set.data[client parameter + 10]++;
   printf("%11d", sum);
```

## Вердикт тестовой системы:



**Вывод:** в ходе выполнения лабораторной работы я закрепил навыки работы со структурами, изучил «простые» способы представления множеств в памяти ЭВМ и создал библиотеки с полезными функциями.