

Программирование

Кузовкина. Е. О

2 февраля 2016 г.

Оглавление

Глава 1

Основные конструкции языка

1.1 Задание 1

1.1.1 Задание 1.1

Пользователь задает моменты начала и конца некоторого промежутка в часах, минутах, секундах, например, 8:20:35 и 13:15:19. Определить длину промежутка в секундах.

1.1.2 Теоретические сведения

Было использовано:

- 1) функции для ввода и вывода информации (scanf, printf - библиотеки `<stdio.h>`)

Для решения данной задачи нужно было использовать стандартную библиотеку ввода и вывода языка C и элементарные математические операции. Для нахождения промежутка времени необходимо было перевести время начала и конца в секунды и найти их разность.

1.1.3 Проектирование

Было выделено две функции:

- 1) `a_gap_of_time.c` - для ввода данных из консоли (взаимодействие с пользователем)
- 2) `numbers.c` - для вычисления промежутка времени (бизнес логика)

1.1.4 Описание тестового стенда и методики тестирования

Использовался Qt Creator 3.5.0 (opensource) с GCC 4.9.1 компилятором
Операционная система: Windows 7

Использовалось ручное тестирование, автоматическое тестирование не проводилось. Ошибок и предупреждений не было.

1.1.5 Тестовый план и результаты тестирования

Все тесты были пройдены успешно: полученный результат, совпал с ожидаемым. Результаты тестирования (например, 7 5 3 - 7 часов, 5 минут, 3 секунды):

1. входные данные - 1 1 1 (начало), 2 2 2 (конец)
ожидаемые значения - 3661
полученные значения - 3661
2. входные данные - 4 4 4 (начало), 4 4 4 (конец)
ожидаемые значения - 0
полученные значения - 0

1.1.6 Выводы

В ходе написания программы не возникло никаких трудностей.

Листинги

```
1 #ifndef __FIRST_TASK_H
2 #define __FIRST_TASK_H
3 #include <stdio.h>
4
5 int hour_beginning, minutes_beginning, seconds_beginning;
6 int hour_end, minutes_end, seconds_end;
7 int beginning_number, end_number, final_number;
8
9 void first_task();
10 int input;
11
12 void a_gap_of_time();
13 int numbers(int beginning_hour, int minutes_beginning, int
    seconds_beginning, int hour_end, int minutes_end, int
    seconds_end);
```

```

14
15 void is_it_a_triangle();
16 int a,b,c;
17
18 void what_type_of_triangle_is_it();
19 int d,e,f;
20 int max;
21
22 #endif // FIRST_TASK

```

```

1 #include "first_task.h"
2
3 void first_task()
4 {
5     printf("Insert a number of the program\n");
6     scanf("%d", &input);
7     switch(input)
8     {
9         case 1:
10             a_gap_of_time();
11             break;
12
13         case 2:
14             is_it_a_triangle();
15             what_type_of_triangle_is_it();
16             break;
17
18         default:
19             printf( "Badly insert\n" );
20     }
21
22 }

```

```

1 #include "first_task.h"
2
3 void a_gap_of_time()
4 {
5     printf("Insert time of the beginning\n");
6     scanf("%d%d%d", &hour_beginning, &minutes_beginning, &
7         seconds_beginning);
8
9     printf("Insert time of the end\n");
10    scanf("%d%d%d", &hour_end, &minutes_end, &seconds_end);
11
12    printf("%d", numbers(hour_beginning, minutes_beginning,
13        seconds_beginning, hour_end, minutes_end, seconds_end)
14    );
15 }

```

```

1 #include "first_task.h"
2
3 int numbers(int hour_beginning, int minutes_beginning, int
  seconds_beginning, int hour_end, int minutes_end, int
  seconds_end)
4 {
5     beginning_number = hour_beginning * 3600 +
        minutes_beginning * 60 + seconds_beginning;
6     end_number = hour_end * 3600 + minutes_end * 60 +
        seconds_end;
7
8     final_number = end_number - beginning_number;
9     return final_number;
10 }

```

1.2 Задание 1

1.2.1 Задание 1.2

Заданы три целых числа: a , b , c . Определить, могут ли они быть длинами сторон треугольника, и если да, является ли данный треугольник остроугольным, прямоугольным или тупоугольным.

1.2.2 Теоретические сведения

Было использовано:

- 1) функции для ввода и вывода информации (`scanf`, `printf` - библиотеки `<stdio.h>`)
- 2) конструкция `"if"`

Для решения данной задачи нужно было использовать стандартную библиотеку ввода и вывода языка C и умение определять вид треугольника. Для этого необходимо было сравнить длину каждой стороны с суммой двух других сторон и сравнить квадрат длины каждой стороны с суммой квадратов двух других сторон.

1.2.3 Проектирование

Было выделено две функции:

- `is_it_a_triangle.c` - для ввода данных из консоли и проверки, существует ли такой треугольник

- `what_type_of_triangle_is_it.c` - для вывода типа треугольника

1.2.4 Описание тестового стенда и методики тестирования

Использовался Qt Creator 3.5.0 (opensource) с GCC 4.9.1 компилятором
Операционная система: Windows 7

Использовалось ручное тестирование, автоматическое тестирование не проводилось. Ошибок и предупреждений не было.

1.2.5 Тестовый план и результаты тестирования

Все тесты были пройдены успешно: полученный результат, совпал с ожидаемым. Результаты тестирования:

1. входные данные - 3 4 5
ожидаемые значения - "Это треугольник. Он правильный"
полученные "This is a right triangle"
2. входные данные - 1 2 3
ожидаемые значения - "Это не треугольник"
полученные "This is not a triangle"

1.2.6 Выводы

При написании данной программы никаких трудностей не было, так как были достаточные знания о геометрических свойствах треугольника.

Листинги

```
1 #include "first_task.h"
2
3 void is_it_a_triangle()
4 {
5     printf("Insert 3 sites of triangle\n");
6     scanf("%d%d%d",&a,&b,&c);
7
8     if ((a >= b+c) || (b >= c+a) || (c >= a+b))
9     {
10         printf("This is not a triangle\n");
11     }
```

```

12     else
13     {
14         printf("This is a triangle\n");
15     }
16 }

```

```

1  #include "first_task.h"
2
3  void what_type_of_triangle_is_it()
4  {
5      printf("Insert 3 sites of triangle\n");
6      scanf("%d%d%d", &d,&e,&f);
7
8      if ((d*d > e*e + f*f) || (e*e > d*d + f*f) || (f*f > d*d
9          + e*e))
10     {
11         printf("This is an obtuse triangle\n");
12     }
13
14     if ((d>e)&&(d>f))
15     {
16         max = d;
17         if ((max*max < e*e + f*f))
18         {
19             printf("This is an acute-angled triangle");
20         }
21     }
22     if ((e>d)&&(e>f))
23     {
24         max = e;
25         if ((max*max < d*d + f*f))
26         {
27             printf("This is an acute-angled triangle");
28         }
29     }
30     else
31     {
32         max = f;
33         if ((max*max < d*d + e*e))
34         {
35             printf("This is an acute-angled triangle");
36         }
37     }
38 }
39
40 if ((d*d == e*e + f*f) || (e*e == d*d + f*f) || (f*f == d
41     *d + e*e))

```



```
42     printf("This is a right triangle\n");  
43 }  
44 }
```

Глава 2

ЦИКЛЫ

2.1 Задание 2

2.1.1 Задание

Сформировать расписание звонков в школе. Время начала занятий 9:00. Количество уроков, длительность урока, длительность малого и большого перерыва, число уроков до большого перерыва (он один раз в день) задаются пользователем. Например, урок 1 2 3 4 5 6 начало 9:00 9:55 10:50 11:45 13:00 13:55 конец 9:45 10:40 11:35 12:30 13:45 14:40

2.1.2 Теоретические сведения

Было использовано:

- 1) функции для ввода и вывода информации (scanf, printf - библиотеки <stdio.h>)
- 2) конструкция "if"

Для решения данной задачи необходимо понимать структуру школьного расписания.

Было создано расписание с учетом перемен.

2.1.3 Проектирование

Было выделено две функции:

- `timetable.c` для ввода данных из консоли (взаимодействие с пользователем)
- `timetable_work.c` для формирования расписания

2.1.4 Описание тестового стенда и методики тестирования

Использовался Qt Creator 3.5.0 (opensource) с GCC 4.9.1 компилятором
Операционная система: Windows 7

Использовалось ручное тестирование, автоматическое тестирование не проводилось. Ошибок и предупреждений не было.

2.1.5 Тестовый план и результаты тестирования

Все тесты были пройдены успешно: полученный результат, совпал с ожидаемым. Результаты тестирования:

1. входные данные - 5 40 10 15 3
ожидаемые - 9:00-9:40, 9:50-10:30, 10:40-11:20, 11:35-12:15, 12:25-13:05
полученные - 9:00-9:40, 9:50-10:30, 10:40-11:20, 11:35-12:15, 12:25-13:05
2. входные данные - 4 50 10 20 2
ожидаемые - 9:00-9:50, 10:00-10:50, 11:20-12:10, 12:10-13:00
полученные - 9:00-9:50, 10:00-10:50, 11:20-12:10, 12:10-13:00

2.1.6 Выводы

В ходе написания программы не возникло никаких трудностей.

Листинги

```
1 #ifndef __TIMETABLE_H
2 #define __TIMETABLE_H
3
4 #include <stdio.h>
5 #include <math.h>
6
7 void timetable(void);
8 int number_of_lessons, length_of_the_lesson, short_break,
   long_break, number_of_lessons_before_a_long_break;
9
10 void timetable_work(int number_of_lessons, int
   length_of_the_lesson, int short_break, int long_break, int
   number_of_lessons_before_a_long_break);
11 int i;
12 int j;
13
14 #endif // TIMETABLE
```

```

1 #include "timetable.h"
2
3 void timetable(void)
4 {
5     printf("Insert a number of lessons\n");
6     scanf("%d", &number_of_lessons);
7
8     printf("Insert length of the lesson\n");
9     scanf("%d", &length_of_the_lesson);
10
11     printf("Insert length of the short break\n");
12     scanf("%d", &short_break);
13
14     printf("Insert length of the long break\n");
15     scanf("%d", &long_break);
16
17     printf("Insert number of lessons before a long break\n");
18     scanf("%d", &number_of_lessons_before_a_long_break);
19
20     timetable_work(number_of_lessons, length_of_the_lesson,
21                   short_break, long_break,
22                   number_of_lessons_before_a_long_break);
23 }

```

```

1 #include "timetable.h"
2
3 void timetable_work(int number_of_lessons, int
4                   length_of_the_lesson, int short_break, int long_break, int
5                   number_of_lessons_before_a_long_break)
6 {
7     j = 9*60;
8     for (i = 1; i <= number_of_lessons; i++)
9     {
10         printf("Lesson:%i\n",i);
11
12         if (j%60 < 10)
13         {
14             printf("Begins:%i:0%i\n", j/60, j%60);
15         }
16
17         else printf("Begins:%i:%i\n", j/60, j%60);
18
19         j += length_of_the_lesson;
20
21         if (j%60 < 10)
22         {
23             printf("Ends:%i:0%i\n", j/60 ,j%60);
24         }
25     }
26 }

```

```
23         }
24
25         else printf("Ends:%i:%i\n", j/60, j%60);
26
27         if (i == number_of_lessons_before_a_long_break)
28         {
29             j += long_break;
30         }
31
32         else
33             j += short_break;
34     }
35 }
```

Глава 3

Массивы

3.1 Задание 3

3.1.1 Задание

Матрицу $K(m,n)$ заполнить следующим образом. Элементам, находящимся на периферии (по периметру матрицы), присвоить значение 1, периметру оставшейся подматрицы - значение 2 и так далее до заполнения всей матрицы.

3.1.2 Теоретические сведения

Было использовано:

- 1) функции для ввода и вывода информации (scanf, printf - библиотеки `<stdio.h>`)
- 2) конструкция "if"
- 3) функции работы с динамической памятью (malloc - библиотеки `<stdlib.h>`)

Для решения данной задачи необходимо уметь работать с динамической памятью.

Нужно было выделить память, построить матрицу, заполненную по определенному принципу.

3.1.3 Проектирование

Было выделено две функции:

- 1) `matrix.c` для ввода данных из консоли (взаимодействие с пользователем)
- 2) `matrix_maker.c` для формирования матрицы

3.1.4 Описание тестового стенда и методики тестирования

Использовался QtCreator с GCC компилятором. Операционная система: Windows 7

Использовалось ручное тестирование, автоматическое тестирование не проводилось. Ошибок и предупреждений не было.

3.1.5 Тестовый план и результаты тестирования

Все тесты были пройдены успешно: полученный результат, совпал с ожидаемым.

Полученные результаты:

1. входные данные - 3 2
 ожидаемые - 1 1 1 1 1 1
 полученные - 1 1 1 1 1 1
2. входные данные - 5 5
 ожидаемые - 1 1 1 1 1 1 2 2 2 1 1 2 3 2 1 1 2 2 2 1 1 1 1 1
 полученные - 1 1 1 1 1 1 2 2 2 1 1 2 3 2 1 1 2 2 2 1 1 1 1 1
3. входные данные - 5 5
 ожидаемые - 1 1 1 1 1 1 1 2 2 2 2 1 1 2 3 3 2 1 1 2 3 3 2 1 1 2 2 2 2 1
 1 1 1 1 1 1
 полученные - 1 1 1 1 1 1 1 2 2 2 2 1 1 2 3 3 2 1 1 2 3 3 2 1 1 2 2 2 2 1
 1 1 1 1 1 1

3.1.6 Выводы

В ходе написания программы не возникло никаких трудностей.

Листинги

```
1 #ifndef __MATRIX_H
2 #define __MATRIX_H
3
4 #include <stdlib.h>
5 #include <stdio.h>
6
7 void matrix();
8 int n;
9 int m;
10
11 int i;
12
13
14 void matrix_maker(int **matrica, int cols, int rows);
15
16 #endif // MATRIX
```

```
1 #include "matrix.h"
2
3 void matrix(void)
4 {
5
6     printf("Insert n\n");
7     scanf("%i",&n);
8
9     printf("Insert m\n");
10    scanf("%i",&m);
11
12    int **matrica=(int **)malloc(m*sizeof(int*));
13
14    for(i = 0; i < m; i++)
15        matrica[i] = (int*) malloc (n*sizeof(int));
16    matrix_maker(matrica, n, m);
17 }
```

```
1 #include "matrix.h"
2
3
4 void matrix_maker(int **matrica, int cols, int rows)
5 {
6     int i;
7     for(i = 0; i < rows; i++)
8     {
9         int j;
10        for(j = 0; j < cols; j++)
11        {
```



```

12         if (i <= j)
13             matrica[i][j] = i + 1;
14         else
15             matrica[i][j] = j + 1;
16     }
17 }
18
19 if (cols >= rows)
20 {
21     int i;
22     for(i = 0; i < rows; i++)
23     {
24         int j;
25         for(j = 0; j < cols; j++)
26         {
27
28             if (i + j >= rows)
29             {
30                 if (i >= j)
31                     matrica[i][j] = rows - i;
32                 else
33                     matrica[i][j] = cols - j;
34             }
35             printf("%i ", matrica[i][j]);
36         }
37
38         printf("\n");
39     }
40 } else {
41     int i;
42     for(i = 0; i < rows; i++)
43     {
44         int j;
45         for(j = 0; j < cols; j++)
46         {
47             if (i + j >= cols)
48             {
49                 if (i > j)
50                     matrica[i][j] = rows - i;
51                 else
52                     matrica[i][j] = cols - j;
53             }
54             printf("%i ", matrica[i][j]);
55         }
56
57         printf("\n");
58     }
59 }
60

```

61 |
62 }

Глава 4

Строки

4.1 Задание 4

4.1.1 Задание

Текст содержит следующие знаки корректуры: \$ - сделать красную строку, # - удалить следующее слово, @ - удалить следующее предложение. Произвести указанную корректировку.

4.1.2 Теоретические сведения

Было использовано:

- 1) функции для ввода и вывода из файла (fopen, fclose- библиотеки <stdio.h>)
- 2) конструкция "if"
- 3) конструкция "while"

Для решения данной задачи необходимо было уметь считывать и записывать информацию в файл, делать корректировку текста. Нужно было считывать текст из файла, проверять его на наличие определенных символов и записывать исправленный текст в файл.

4.1.3 Проектирование

The text of this article is in English. \$ See Russian Names in English (Russian Text) for the text in Russian.

Different ways of rendering Russian names into English existed in the past, and several standards of transliteration of Cyrillic into English exist

now. As a result, there may be several English spelling variants for the same Russian name or surname: Yulia, Yuliya, Julia, Julja (Юлия); Dmitry, Dmitriy, Dmitri, Dimitri (Дмитрий); Yevgeny, Yevgeniy, Evgeny, Evgeni, Evgeniy, Eugeny (Евгений); Tsvetaeva, Tsvetayeva, Cvetaeva (Цветаева); Zhukovsky, Zhukovski, Zhukovskiy, Jukovsky (Жуковский).

In some cases, the number \$ of existing English \$ variants is really intimidating. For example, # Муравьев, a common Russian last name, is represented by more than fifteen variants of spelling in English: Muravyov, Myravyev, Muraviev, Muraviov, Murav'ev, Muravev, Murav'yev, Murav'ov, Muravjov, Muravjev, Mouravieff, Muravieff, Mouravief, Muravief, Muraviof, Muravioff.

Generally, # transliteration of Russian names into Latin is English-oriented now. @ But many Russian names were transliterated according to the French language in the past, and transliteration on the basis of French was the norm for names and surnames in our travel passports until recently. As a result, French-oriented transliteration variants of Russian names are still common. Also, English spelling of Russian names is influenced by tradition and people's personal preferences.

Modern ways of rendering Russian names into English \$ try to preserve, as much as possible, both the pronunciation and the recognizable written look of the original Russian name. This material offers examples of typical English spelling variants for Russian names and linguistic recommendations for rendering Russian names into English. Bear in mind that your name should be written in the same way in all of your travel documents because discrepancies may lead to problems when travelling.

4.1.4 Описание тестового стенда и методики тестирования

Использовался QtCreator с GCC компилятором
Операционная система: Windows 7
Использовалось ручное тестирование, автоматическое тестирование не проводилось. Ошибок и предупреждений не было.

4.1.5 Тестовый план и результаты тестирования

Все тесты были пройдены успешно: полученный результат, совпал с ожидаемым, то есть была произведена корректировка текста.

Полученный текст: The text of this article is in English. See Russian Names in English (Russian Text) for the text in Russian.

Different ways of rendering Russian names into English existed in the past, and several standards of transliteration of Cyrillic into English exist

now. As a result, there may be several English spelling variants for the same Russian name or surname: Yulia, Yuliya, Julia, Julja (Юлия); Dmitry, Dmitriy, Dmitri, Dimitri (Дмитрий); Yevgeny, Yevgeniy, Evgeny, Evgeni, Evgeniy, Eugeny (Евгений); Tsvetaeva, Tsvetayeva, Svetaeva (Цветаева); Zhukovsky, Zhukovski, Zhukovskiy, Jukovsky (Жуковский).

In some cases, the number of existing English variants is really intimidating. For example, Муравьев, a common Russian last name, is represented by more than fifteen variants of spelling in English: Muravyov, Myravyev, Muraviev, Muraviov, Murav'ev, Muravev, Murav'yev, Murav'ov, Muravjov, Muravjev, Mouravieff, Muravieff, Mouravief, Muravief, Muraviof, Muravioff.

Generally, transliteration of Russian names into Latin is English-oriented now. . As a result, French-oriented transliteration variants of Russian names are still common. Also, English spelling of Russian names is influenced by tradition and people's personal preferences.

Modern ways of rendering Russian names into English try to preserve, as much as possible, both the pronunciation and the recognizable written look of the original Russian name. This material offers examples of typical English spelling variants for Russian names and linguistic recommendations for rendering Russian names into English. Bear in mind that your name should be written in the same way in all of your travel documents because discrepancies may lead to problems when travelling.

4.1.6 Выводы

В ходе написания программы не возникло никаких трудностей.

Листинги

```
1 #include <stdio.h>
2
3 void read_word(char *s, int *i, char *w) //чтение слова
4 {
5     int j;
6     while (s[*i] <= ' ')
7         (*i)++;
8     j = 0;
9     while (s[*i] != '\0' && s[*i] != ' ')
10    {
11        w[j] = s[*i];
12        j++;
13        (*i)++;
14    }
15    w[j] = '\0';
```

```

16 }
17 void strings()
18 {
19     FILE *f,*g;
20     char s[2000], w[2000], ws[2000], pred[2000], pred1[2000];
21     s[0] = '\0';
22     ws[0] = '\0';
23     w[0] = '\0';
24     int vsp=1;
25     int sl = 0;
26     int str;
27     int j = 0;
28     f = fopen("2.txt", "r");
29     if (!f)
30     {
31         puts("cannot open the file");
32         return;
33     }
34     while (!feof(f))
35     {
36         fgets(s, 2000, f);
37
38         int i = 0;
39         str = 0;
40         ws[0] = '\0';
41         if (vsp)
42             while (s[i] != '\0')
43             {
44                 if (s[i] == '$')
45                 {
46                     w[j] = '\n';
47                     j++;
48                     w[j] = ' ';
49                     j++;
50                     w[j] = ' ';
51                     j++;
52                     w[j] = '\0';
53                     i++;
54                 }
55                 else
56                 {
57                     if (s[i] == '#')
58                         read_word(s, &i, ws);
59                     else
60                     {
61                         if (s[i] == '@')
62                         {
63                             while (s[i + 1] != '.' && s[i + 1] != '

```

```

64         \0')
65         i++;
66         i++;
67         if (s[i] == '\0')
68             vsp = 0;
69     }
70     else
71     {
72         w[j] = s[i];
73         i++;
74         j++;
75         w[j] = '\0';
76     }
77 }
78 }
79 else
80 {
81     while (s[i + 1] != '.' && s[i + 1] != '!' && s[i + 1]
82            != '?' && s[i + 1] != '\0')
83         i++;
84     i++;
85     if (s[i] != '\0')
86         vsp = 1;
87 }
88 }
89 printf("text:\n");
90 puts(w);
91 fclose(f);
92 remove("2.txt");
93 g = fopen("2.txt", "w");
94 fputs(w, g);
95 fclose(g);
96 getchar();
97 getchar();
98 }

```

Глава 5

Задание на классы

5.1 Множество

5.1.1 Задание 1

Множество Реализовать класс МНОЖЕСТВО (целых чисел). Требуемые методы: конструктор, деструктор, копирование, сложение множеств, пересечение множеств, добавление в множество, включение в множество.

5.1.2 Теоретические сведения

Множество - совокупность некоторых объектов (элементов).

Объединение множеств - образование нового множества содержащего все элементы обоих множеств, без повторений элементов.

Равные множества - множества, содержащие одинаковые элементы.

Пересечение множеств - образуется множество, содержащее только элементы принадлежащие обоим множествам.

Подмножество множества - часть множества.

Разность множеств - множество, содержащее элементы одного множества, но не содержащее элементы другого. Было использовано:

- 1) потоки ввода и вывода информации - (iostream, ostream - библиотеки <stdlib>)
- 2) конструкция "if"
- 3) циклы "while" и "for"
- 4) классы

Для решения данной задачи необходимо уметь работать с классами и потоками ввода и вывода. Нужно было создать класс и его методы.

5.1.3 Проектирование

В классе были реализованы следующие поля:

- 1) mas - массив, представляющий из себя множество
- 2) k - переменная, используемая в качестве счетчика

и следующие методы:

- 1) Add - функция, которая добавляет элементы в множество
- 2) Sub - функция, которая убирает элементы из множества
- 3) HowMany - функция, которая возвращает количество элементов в множестве

Также были перегружены следующие операторы:

- 1) "+=" оператор, использующий функцию Add, добавляет элемент в множество или объединяет два множества
- 2) "=" оператор, использующий функцию Sub, убирает элемент из множества или вычитает одно множество из другого множества
- 3) "*=" пересечение двух множеств
- 4) "-=" логический оператор, проверяющий равны ли два множества
- 5) "<=" логический оператор, проверяющий является ли одно множество подмножеством другого
- 6) "<< >>" ввод и вывод в поток

5.1.4 Описание тестового стенда и методики тестирования

Использовался QtCreator с GCC компилятором
Операционная система: Windows 7
Использовалось ручное тестирование, автоматическое тестирование не проводилось. Ошибок и предупреждений не было.

5.1.5 Тестовый план и результаты тестирования

Все тесты были пройдены успешно: полученный результат, совпал с ожидаемым.

5.1.6 Выводы

В ходе написания программы не возникло никаких трудностей

Листинги

```
1 #include <iostream>
2 #include <stdio.h>
3 #include <stdlib.h>
4
5 using namespace std;
6
7 class Array
8 {
9     int *mas, k;
10    void Add(int);
11    void Sub(int);
12 public:
13     Array():k(0), mas(new int(0)) {};
14     Array(const Array &);
15     ~Array() {delete [] mas;};
16     void operator+=(int n) {Add(n);};
17     void operator+=(Array &);
18     void operator-=(int n) {Sub(n);};
19     void operator-=(const Array &);
20     Array operator*(const Array &) const;
21     void operator*=(const Array &);
22     friend bool operator==(Array &, Array &);
23     friend bool operator<=(Array &, Array &);
24     friend ostream& operator<<(ostream &, const Array &);
25     friend istream& operator>>(istream &s, Array &);
26     int HowMany() {return k;};
27 };
28
29
30 Array::Array(const Array &x):k(x.k)
31 {
32     mas = new int[k];
33     for(int i = 0; i < k; i++)
34         mas[i] = x.mas[i];
35 }
36
37 void Array::Add(int n)
38 {
39     int *t, pos;
40     for (pos = 0; pos < k && mas[pos] < n; pos++) {}
41     if (mas[pos] != n)
42     {
```

```

43         t=new int[++k];
44         for(int i = 0; i < k-1; i++)
45             t[i < pos?i:i+1] = mas[i];
46         t[pos] = n;
47         delete []mas;
48         mas = t;
49     }
50 }
51
52 void Array::operator+=(Array &x)
53 {
54     for(int i = 0; i < x.k; i++)
55         Add(x.mas[i]);
56 }
57
58 void Array::Sub(int n)
59 {
60     if (k > 0)
61     {
62         int *t, pos;
63         for (pos = 0; pos < k && mas[pos] < n; pos++) {}
64         if (mas[pos] == n)
65         {
66             t=new int[--k];
67             for(int i = 0; i < k+1; i++)
68                 if (i != pos) t[i < pos?i:i-1] = mas[i];
69             delete []mas;
70             mas = t;
71         }
72     }
73 }
74
75 void Array::operator-=(const Array &x)
76 {
77     for (int i = 0; i < x.k; i++)
78         return Sub(x.mas[i]);
79 }
80
81 Array Array::operator*(const Array &x) const
82 {
83     Array t(*this), t2(*this);
84     t-=x;
85     for(int i = 0; i < t.k; ++i)
86         t2.Sub(t.mas[i]);
87     return t2;
88 }
89
90 void Array::operator*=(const Array &x)
91 {

```

```

92     Array t(*this);
93     t-=x;
94     for (int i = 0;i<t.k;i++)
95         Sub(t.mas[i]);
96 }
97
98 bool operator==(Array &x, Array &y)
99 {
100     if (x.k != y.k) return false;
101     for (int i = 0; i < x.k; i++)
102         if (x.mas[i] != y.mas[i]) return false;
103     return true;
104 }
105
106 bool operator<=(Array &x, Array &y)
107 {
108     int s = 0;
109     for (int i = 0; i < x.k; i++)
110         while(x.mas[i] != y.mas[i+s])
111         {
112             if (x.k+s > y.k) return false;
113             s++;
114         }
115     return true;
116 }
117
118 ostream &operator<<(ostream &s, const Array &p)
119 {
120     if (p.k != 0){
121         s<<" ";
122         for (int i = 0;i < p.k-1; i++){
123             s<<p.mas[i]<<" ";
124         }
125         s<<p.mas[p.k-1];
126     }
127     return s<<" ";
128 }
129 istream &operator>>(istream &s, Array &p)
130 {
131     int tmp;
132     char c;
133     s>>c;
134
135     while(c!='')
136     {
137         s>>tmp>> c;
138         p.Add(tmp);
139     }
140     return s;

```

141
142

}