Олимпиада школьников «Шаг в будущее» по общеобразовательному предмету "Информатика". 2014 год. Отборочный этап. 10-11 классы. Билет 2.

Задача 1: Ряд Фибоначчи (10)

Последовательность чисел 1, 1, 2, 3, 5, 8, 13, 21, …, a_n, …, два первых члена которой равны 1, а каждый член, начиная с третьего, равен сумме двух предыдущих: $a_{n+2} = a_{n+1} + a_n$, называется «рядом» Фибоначчи. Вычислить последнюю цифру числа a_{15k} (k – целое).

Входные данные. Во входном файле записано одно целое число k ($1 \le k \le 10^9$).
Выходные данные. В выходной файл вывести одно целое число – последнюю цифру искомого числа.

Пример входного файла	Пример выходного файла
1	0

РЕШЕНИЕ:

```
#include "stdafx.h"

FILE *ifs, *ofs;

int _tmain(int argc, _TCHAR* argv[])
{
	unsigned long long k;
	ifs = fopen( "1.IN", "r" );
	fscanf( ifs, "%lld", &k );
	fclose( ifs );
	ofs = fopen( "1.OUT", "w" );
	fprintf( ofs, "%d\n", 0 ); // всегда 0
	fclose( ofs );
	return 0;
}
```

Задача 2: Сумма парных произведений (15)

Найти сумму парных произведений n натуральных чисел, т. е. сумму:

1*2 + 1*3 + … + 1*n + 2*3 + 2*4 + … + 2*n + … + (n-1)*n.

Входные данные. Во входном файле записано одно целое число n ($2 \le n \le 10^4$).
Выходные данные. В выходной файл вывести одно целое число – значение вычисленной суммы.

Пример входного файла	Пример выходного файла
10	1320
100	12582075

РЕШЕНИЕ:

```
#include "stdafx.h"

FILE *ifs, *ofs;

int _tmain(int argc, _TCHAR* argv[])
{
	unsigned long long n;
	ifs = fopen( "2.IN", "r" );
	fscanf( ifs, "%lld", &n );
	fclose( ifs );
	ofs = fopen( "2.OUT", "w" );
// Аналитическое решение
	fprintf( ofs, "%lld\n", (n-1)*n*(n+1)*(3*n+2)/24);
// Алгоритмическое решение
	unsigned long long i, j, s1, s2;
	s1 = 0;
	for (i=1; i<n; i++)
	{
```

```
            s2 = 0;
            for (j=i+1; j<n+1; j++)
                  s2+=i*j;
            s1+=s2;
      }
      fprintf( ofs, "%lld\n", s1);
      return 0;
}
```

Задача 3: Текст (20)

Дан текст на английском языке. Предложения в тексте начинаются с заглавной буквы, а в конце предложения ставится "." (точка). Слова в предложениях и сами предложения отделяются друг от друга, по крайней мере, одним пробелом. Между словами допускаются символы пунктуации: "," (запятая), ";" (точка с запятой), ":" (двоеточие), "–" (тире). Символ апострофа считается частью слова. Получить 5 наиболее часто встречающихся слов и число их появлений.

Входные данные. Во входном файле записан текст на английском языке. Длина строки текста не более 60. Количество строк текста не более 1000. Длина слова не более 20.
Выходные данные. Слова должны быть напечатаны в верхнем регистре. Каждое слово и его количество печатать с новой строки.

Пример входного файла	Пример выходного файла
I'll tell you a story, a story anon, Of a noble prince, and his name was King John. For he was a prince, and a prince of great might, He held up great wrongs, he put down great right.	A 5 PRINCE 3 HE 3 GREAT 3 STORY 2

РЕШЕНИЕ:

```
#include "stdafx.h"
#include <stdlib.h>
#include <string.h>

FILE *ifs, *ofs;

typedef struct tnode TNODE;

struct tnode {
      char word[21];
      int count;
      int flag;
      TNODE *right;
};

void add( char w[21], TNODE *&p );
void print( TNODE *p );

int _tmain(int argc, _TCHAR* argv[])
{
      char line[100], *copy, *token;
      char seps[] = " .,;:-\n\r";
      TNODE *list = NULL;
      ifs = fopen( "3.IN", "r");
      ofs = fopen( "3.OUT", "w");
      while (!feof( ifs ))
      {
            if (fgets( line, 100, ifs ) != NULL)
            {
                  copy = strupr( strdup( line ) );
                  token = strtok( copy, seps );
                  while (token != NULL)
                  {
                        add( token, list );
                        token = strtok( NULL, seps );
                  }
            }
```

```
	}
	print( list );
	fclose( ifs );
	fclose( ofs );
	return 0;
}

void add( char w[21], TNODE *&p )
{
	TNODE *curr, *last, *temp;
	int flag = 0;
	if (p == NULL)
	{
		p = (TNODE *)malloc( sizeof( TNODE ) );
		strcpy( p->word, w );
		p->count = 1;
		p->flag = 0;
		p->right = NULL;
	}
	else
	{
		curr = p;
		while (curr != NULL)
		{
			if (strcmp( curr->word, w ) == 0)
			{
				curr->count++;
				flag = 1;
				break;
			}
			last = curr;
			curr = curr->right;
		}
		if (flag == 0)
		{
			temp = (TNODE *)malloc( sizeof( TNODE ) );
			strcpy( temp->word, w );
			temp->count = 1;
			temp->flag = 0;
			temp->right = NULL;
			last->right = temp;
		}
	}
}

void print( TNODE *p )
{
	/*if (p == NULL) printf("LIST IS EMPTY\n");*/
	TNODE *curr, *pmax;
	int i;
	int countmax;
	for (i = 0; i < 5; i++)
	{
		pmax = NULL;
		countmax = 0;
		curr = p;
		while (curr != NULL)
		{
			if (curr->flag == 0)
			{
				if (curr->count > countmax)
				{
					countmax = curr->count;
					pmax = curr;
				}
			}
			curr = curr->right;
```

```
            }
            if (pmax != NULL)
            {
                  fprintf( ofs, "%s\t%d\n", pmax->word, pmax->count);
                  pmax->flag = 1;
            }
      }
}
```

Задача 4: Выражение (25)

Обычный метод записи математических выражений, в которых бинарный оператор записывается между операндами, известен под названием инфиксной записи. При отсутствии скобок операции выполняются согласно правилам приоритета операторов. Для изменения порядка выполнения операций применяют скобки. Постфиксная польская запись – это форма записи математических выражений, в которой операнды расположены перед оператором. Если оператор имеет фиксированную арность, то в такой записи будут отсутствовать скобки, и она может быть интерпретирована без неоднозначности. Для заданного инфиксного логического выражения получить последовательность операторов в постфиксной польской записи.

Входные данные. Входной файл содержит одну строку, в которой записано инфиксное логическое выражение. В исходном выражении нет пробелов, в качестве операндов используются латинские буквы в верхнем или нижнем регистре, в качестве операторов используются знаки "~" (отрицание), "&" (конъюнкция), "|" (дизъюнкция). Длина каждого выражения не превосходит 100.
Выходные данные. В выходной файл вывести одну строку, в которой записаны операторы постфиксной польской записи.

Пример входного файла	Пример выходного файла
x&(~x|~~y)&(z|y)	~~~|&|&

РЕШЕНИЕ:

```
#include "stdafx.h"
#include <ctype.h>
#include <string.h>

const int N = 100;

FILE *ifs, *ofs;

static int op_stack[N+1];
static int ex_stack[N+1];
static int op_index = 0, ex_index = 0;

void op_push( int value )
{
      op_stack[++op_index] = value;
}

int op_pop()
{
      op_index--;
      return (op_stack[op_index+1]);
}

int op_top()
{
      return (op_stack[op_index]);
}

void ex_push( int value )
{
      ex_stack[++ex_index] = value;
}

int is_operator( int c )
{
      if (c == '|' || c == '&' || c == '~' ) return 1;
```

```
      else return 0;
}

int op_priority( int op )
{
      if (op == '~') return 4;
      if (op == '&') return 3;
      if (op == '|') return 2;
      return 1;
}

int _tmain(int argc, _TCHAR* argv[])
{
      char line[N+1];
      int m, i;
      char ch, top;
      // Читаем строку выражения
      ifs = fopen( "4.IN", "r" );
      fgets( line, N, ifs );
      fclose( ifs );
      m = strlen( line );
      if (m == 0) return 1;
      // Реализуем алгоритм сортировочной станции
      op_stack[0] = 0;
      ex_stack[0] = 0;
      for (i = 0; i < m; i++)
      {
            ch = line[i];
            if (isalpha( ch )) ex_push( ch );
            else if (ch == '(') op_push( ch );
            else if (ch == ')')
            {
                  top = op_top();
                  while (top != '(' && op_index != 0)
                  {
                        ex_push( top );
                        op_pop();
                        top = op_top();
                  }
                  if (top == '(') op_pop();
            }
            else if (is_operator( ch ))
            {
                  top = op_top();
                  while (is_operator( top ) && op_priority( top ) >= op_priority( ch )
&& op_index != 0)
                  {
                        ex_push( top );
                        op_pop();
                        top = op_top();
                  }
                  op_push( ch );
            }
            else;
      }
      top = op_top();
      while (is_operator( top ) && op_index != 0)
      {
            ex_push( top );
            op_pop();
            top = op_top();
      }
      // Печатаем результат
      ofs = fopen( "4.OUT", "w" );
      for (i = 1; i <= ex_index; i++)
            if (!isalpha( ex_stack[i] ))
                  fprintf( ofs, "%c", ex_stack[i] );
```

```
	fprintf( ofs, "\n" );
	fclose( ofs );
	return 0;
}
```

Задача 5: Вложенные матрешки (30)

Матрёшки представляют собой вложенные друг в друга деревянные куклы. Пусть каждая матрешка имеет форму эллипсоида вращения с полуосями a_i (ширина матрешки) и b_i (высота матрешки). Условием вложения i-ой матрешки в j-ю матрешку является выполнение неравенств $a_i < a_j$ и $b_i < b_j$. Найти наименьшее количество оставшихся матрешек после их укладки.

Входные данные. В первой строке входного файла содержится целое число n – количество матрешек ($1 \le n \le 20000$). Во второй строке входного файла записано $2*n$ целых положительных чисел $a_1, b_1, a_2, b_2, \ldots, a_n, b_n$, где $1 \le a_i, b_i \le 10^4$ для всех i.
Выходные данные. В выходной файл вывести одно целое число – наименьшее количество оставшихся матрешек.

Примеры входного файла	Примеры выходного файла
3 20 30 40 50 30 40	1
4 20 30 10 10 30 20 40 50	2
3 10 30 20 20 30 10	3
4 10 10 20 30 40 50 39 51	2

РЕШЕНИЕ:

```
#include "stdafx.h"
#include <search.h>

const int MAXM = 20001;

struct dim // Размер матрешки
{
	int w;		// Ширина матрешки
	int h;		// Высота матрешки
};

FILE *ifs, *ofs;

struct dim dims[MAXM];  // Массив матрешек
int size[MAXM+1];
int anti_chain_size;
int compare( const void *arg1, const void *arg2 ); // Прототип компаратора для функции
qsort

int _tmain(int argc, _TCHAR* argv[])
{
	int m, i, j, lo, hi, mid;
	// Читаем исходные данные
	ifs = fopen( "5.IN", "r" );
	fscanf( ifs, "%d", &m);
	for (i = 0; i < m; i++)
		fscanf( ifs, "%d %d", &dims[i].w, &dims[i].h );
	fclose( ifs );
	// Решаем задачу
	qsort( dims, m, sizeof(struct dim), compare );
	anti_chain_size = 0;
	for (i = 0; i < m; i++)
	{
		hi = anti_chain_size;
		lo = 0;
		while (hi > lo)
		{
			mid = (hi + lo)/2;
```

```
                if (size[mid] >= dims[i].h)
                        lo = mid + 1;
                else
                        hi = mid;
            }
            size[lo] = dims[i].h;
            anti_chain_size += (lo == anti_chain_size);
      }
      // Печатаем ответ
      ofs = fopen( "5.OUT", "w" );
      fprintf( ofs, "%d\n", anti_chain_size );
      fclose( ofs );
      return 0;
}

int compare( const void *arg1, const void *arg2 )
{
      struct dim* t1 = ((struct dim*)arg1);
      struct dim* t2 = ((struct dim*)arg2);
      if (t1->w != t2->w) return t1->w - t2->w;
      return t2->h - t1->h;
}
```