پروژه برنامه نویسی – درس اصول طراحی کامپایلر

صورت مسئله:

عنوان: تحلیلگر لغوی

توی این برنامه قراره یک تحلیلگر لغوی طراحی کنیم که نوع‌هایی (Type) از اعداد و رشته‌ها رو به شکل زیر شناسایی کنه:

Token	Example	Informal description
unsigned integer	1, 5125, 315, . .	all sequences of digits
signed integer	+6, -152, . . .	all sequences of digits starting with + or -
decimal number	10.151, 3.1415, 0.0001, .. .	includes a decimal point and at least one digit to the left and right of the point. Possibly starts with + or -
scientific number	10e-01, 43.21e+10, 21e10,. . .	a decimal number or a (signed, unsigned) integer followed by e and a signed or unsigned integer.
Name	a512, key, ch153, counter, . . .	starts with a letter.
String	1a512, 61a, 12df as0, 01basdf	starts with a digit but contains also other letters (and is not a scientific number)
Relation		< or > or = or <= or >= or ! =
unknown		any other character (errors)

برنامه باید فضاهای خالی رو حذف کنه؛ همینطور تعداد اعداد، رشته‌ها و نام‌ها رو محاسبه و چاپ کنه. همینطور برای هر lexeme، شماره‌ی خط متناظر باهاش و token شناسایی شده رو چاپ کنه.

همینطور برنامه باید تمام اعدادی رو که پیدا کرده داخل جدول اعداد (NumTable) که ساختارش به شکل زیر هست ذخیره کنه و در انتهای برنامه جمع کل اعداد رو محاسبه و چاپ کنه. اندازه‌ی جدول هم باید ثابت و به اندازه‌ی ۱۰۰ درنظر گرفته بشه.

ساختار جدول باید به شکل زیر باشه:

```
typedef struct {
double num;
int lineNum;
} NumRecord;
NumRecord numTable[MAX_COUNT];
double sum = 0;
```

برای مثال، حاصل اجرای برنامه روی فایل ورودی زیر، باید به شکلی که در ادامه اومده باشه:

File:
12.3 local 21fa43
33.5e+3 x<=name
-44 +1520 a=3

should be processed as:
12.3, line1, decimal number
local, line1, name
21fa43, line1, string
33.5e+3, line2, scientific number
x, line2, name
<=, line2, relation
name, line2, name
-44, line3, signed integer
+1520, line3, signed integer
a, line3, name
=, line3, relation
3, line3, unsigned integer

5 numbers found.
SUM = 34991.300000

Contents of Number Table

0	L1	12.300000
1	L2	33500.000000
2	L3	-44.000000
3	L3	1520.000000
4	L3	3.000000

توضیح پروژه:

یه سایز ثابت به اندازه‌ی ۱۰۰ برای مشخص کردن سایز آرایه‌هامون مشخص می‌کنیم:

```
const int MAX_COUNT = 100;
```

یه struct به صورت زیر می‌سازیم که داخلش دوتا متغیر هست. یکی به اسم num که مقدار عددها داخلش ذخیره میشه و از نوع double هست، و یکی به اسم lineNum از نوع int که شماره‌ی خط رو مشخص می‌کنه

```
typedef struct NumberTable {...}
```

تابع ShowHeader فقط یک تابع پرینت ساده س :)

```
void ShowHeader() {...}
```

این تابع مسیری که فایل ورودی‌تون داخلش هست رو برمیگردونه

```
string LoadFilePath() {...}
```

این تابع هر سطر از فایل شما رو می‌گیره و tokenهای داخل هر سطر رو جدا میکنه و توی آرایه‌ی lexeme میریزه

```
void SplitLexemes(string path, string lexemes[MAX_COUNT][2]) {...}
```

این تابع tokenهای جدا شده رو بررسی می‌کنه و نوعش رو مشخص می‌کنه. همینطور اعداد رو هم شناسایی می‌کنه و داخل numTable میریزه

```
void FindToken(string lexemes[MAX_COUNT][2], NumRecord numTable[MAX_COUNT]) {...}
```

این تابع اعداد رو نمایش میده و جمع اعداد رو حساب می‌کنه و همینطور اونا رو چاپ می‌کنه

```
void ShowNumbers(NumRecord numTable[MAX_COUNT]) {...}
```

اینم main برنامه که از توابع بالا استفاده می‌کنه :)

```
int main()
{
	ShowHeader();
	string filePath = LoadFilePath();
	string lexemes[MAX_COUNT][2];
	NumRecord numTable[MAX_COUNT];

	SplitLexemes(filePath, lexemes);
	if (lexemes[0][0] == "Error") // if file doesn't exist or content is empty
	{
		cout << "\n" << lexemes[0][1];
	}
	else
	{
		FindToken(lexemes, numTable);
		ShowNumbers(numTable);
	}
	_getch();
	return 0;
}
```