

Разбор задания №3

Римский математик

Постановка задачи

Вы работаете в компании, разрабатывающей цифровые решения для музеев, архивов и библиотек, специализирующихся на исторических документах. В таких учреждениях по-прежнему используется римская система счисления при описании дат, инвентарных номеров и других числовых значений, чтобы сохранить историческую аутентичность документов.

Необходимо разработать алгоритм, способный выполнять базовые арифметические операции с римскими числами, так как сотрудники часто сталкиваются с задачами вычислений в этой системе. Алгоритм должен уметь:

- Складывать два римских числа;
- Вычитать одно римское число из другого;
- Умножать два римских числа.

Постановка задачи

Формат входных данных:

На вход программа принимает два римских числа (от I до MMMCMXCIX, т.е. от 1 до 3999) и оператор (*, '-', '+'). Все данные вводятся в одной строке слитно.

Гарантируется, что если римское число введено, то оно введено правильно.

Формат выходных данных:

- Программа выводит результат операции в римской системе счисления.
- Если вход содержит недопустимые символы, программа должна вывести **'ERROR'**.
- Если результат вычисления выходит за пределы допустимых римских чисел (меньше I или больше MMMCMXCIX), также вывести **'ERROR'**.

Примеры входных и выходных данных

Ввод	Вывод	Объяснение
III+XLIX	LII	$3 \text{ (III)} + 49 \text{ (XLIX)} = 52 \text{ (LII)}$
XX-XXX	ERROR	$20 \text{ (XX)} - 30 \text{ (XXX)} = -10$, операция некорректна так как римскими числами нельзя записать отрицательные числа и ноль
AB+X	ERROR	Введены некорректные символы

Заготовка

```
FUNCTION RomanValue(Symbol: CHAR): INTEGER;  
BEGIN  
    IF Symbol = 'I' THEN RomanValue := 1 ELSE  
    IF Symbol = 'V' THEN RomanValue := 5 ELSE  
    IF Symbol = 'X' THEN RomanValue := 10 ELSE  
    IF Symbol = 'L' THEN RomanValue := 50 ELSE  
    IF Symbol = 'C' THEN RomanValue := 100 ELSE  
    IF Symbol = 'D' THEN RomanValue := 500 ELSE  
    IF Symbol = 'M' THEN RomanValue := 1000  
END;
```

Заготовка

```
PROCEDURE PrintRoman(Number: INTEGER);
BEGIN
    WHILE Number >= 1000 DO BEGIN WRITE('M'); Number := Number - 1000 END;
    IF Number >= 900 THEN BEGIN WRITE('CM'); Number := Number - 900 END;
    IF Number >= 500 THEN BEGIN WRITE('D'); Number := Number - 500 END;
    IF Number >= 400 THEN BEGIN WRITE('CD'); Number := Number - 400 END;
    WHILE Number >= 100 DO BEGIN WRITE('C'); Number := Number - 100 END;
    IF Number >= 90 THEN BEGIN WRITE('XC'); Number := Number - 90 END;
    IF Number >= 50 THEN BEGIN WRITE('L'); Number := Number - 50 END;
    IF Number >= 40 THEN BEGIN WRITE('XL'); Number := Number - 40 END;
    WHILE Number >= 10 DO BEGIN WRITE('X'); Number := Number - 10 END;
    IF Number >= 9 THEN BEGIN WRITE('IX'); Number := Number - 9 END;
    IF Number >= 5 THEN BEGIN WRITE('V'); Number := Number - 5 END;
    IF Number >= 4 THEN BEGIN WRITE('IV'); Number := Number - 4 END;
    WHILE Number >= 1 DO BEGIN WRITE('I'); Number := Number - 1 END;
    WRITELN
END;
```

Анализ заготовки

Что мы имеем:

- Пройдет тест, если ответ был 5
- Мы умеем получать значение некоторых римских цифр
- Мы можем вывести любое римское число

Но:

- Нет обработки ошибок
- Не реализован сам алгоритм работы с римскими числами

Можно сделать лучше

Добавляем константы и глобальные переменные

CONST

```
NotRomanNumber = -1;  
NotOperation = ' ';
```

VAR

```
FirstRoman, SecondRoman, ResultValue: INTEGER;  
Operation: CHAR;  
Error: BOOLEAN;
```

Добавляем обработку неправильной записи

```
FUNCTION RomanValue(Symbol: CHAR): INTEGER;  
BEGIN  
    IF Symbol = 'I' THEN RomanValue := 1 ELSE  
    IF Symbol = 'V' THEN RomanValue := 5 ELSE  
    IF Symbol = 'X' THEN RomanValue := 10 ELSE  
    IF Symbol = 'L' THEN RomanValue := 50 ELSE  
    IF Symbol = 'C' THEN RomanValue := 100 ELSE  
    IF Symbol = 'D' THEN RomanValue := 500 ELSE  
    IF Symbol = 'M' THEN RomanValue := 1000 ELSE  
        RomanValue := NotRomanNumber  
END;
```

Добавляем основной блок программы

```
BEGIN
  ReadParameters(Operation, FirstRoman, SecondRoman, Error);
  ResultValue := GetResult(Operation, FirstRoman, SecondRoman, Error);
  IF NOT Error
  THEN
    PrintRoman(ResultValue)
  ELSE
    WRITELN('ERROR')
  END.
```

Добавляем функцию считывания параметров

```
PROCEDURE ReadParameters(VAR Operation: CHAR; VAR FirstRoman, SecondRoman: INTEGER; VAR Error: BOOLEAN);
BEGIN
    Operation := NotOperation;
    Error := FALSE;
    FirstRoman := ReadRoman(Operation);
    IF (FirstRoman = NotRomanNumber) OR (Operation = NotOperation)
    THEN
        Error := TRUE;
    SecondRoman := ReadRoman(Operation);
    IF SecondRoman = NotRomanNumber
    THEN
        Error := TRUE
    END;
END;
```

```

FUNCTION ReadRoman(VAR Operation: CHAR): INTEGER;
VAR
    Previous, Current, ReturnValue: INTEGER;
    Symbol: CHAR;
BEGIN
    ReturnValue := NotRomanNumber;
    Previous := 0;
    WHILE NOT EOLN
    DO
        BEGIN
            READ(Symbol);
            IF (Symbol = '+') OR (Symbol = '-') OR (Symbol = '*')
            THEN
                BEGIN
                    Operation := Symbol;
                    ReadRoman := ReturnValue;
                    EXIT
                END;
            Current := RomanValue(Symbol);
            IF Current = NotRomanNumber
            THEN
                BEGIN
                    ReadRoman := NotRomanNumber;
                    EXIT
                END;
            IF ReturnValue = NotRomanNumber
            THEN
                ReturnValue := 0;
            IF Current > Previous
            THEN
                ReturnValue := ReturnValue + Current - 2 * Previous
            ELSE
                ReturnValue := ReturnValue + Current;
            Previous := Current
        END;
    ReadRoman := ReturnValue
END;

```

Добавляем функцию
считывания римского числа

Добавляем функцию получения результата

```
FUNCTION GetResult(Operation: CHAR; FirstRoman, SecondRoman: INTEGER; VAR Error: BOOLEAN): INTEGER;  
VAR  
    Res: INTEGER;  
BEGIN  
    IF Operation = '+' THEN Res := FirstRoman + SecondRoman ELSE  
    IF Operation = '-' THEN Res := FirstRoman - SecondRoman ELSE  
    IF Operation = '*' THEN Res := FirstRoman * SecondRoman ELSE  
        Error := TRUE;  
    IF (Result < 1) OR (Result > 3999)  
    THEN  
        Error := TRUE;  
    GetResult := Res  
END;
```

Что изменилось

Теперь мы:

- Считываем римские числа
- Проводим над римскими числами операции умножения, вычитания и сложения
- Обрабатываем ошибки

Вопросы