Блок-6 Машины (обязательные) по теме «Рекурсия, часть 2»

(всего 6 задач, срок их сдачи до 3 декабря включительно):

“Без троек”. Написать программу, которая вводит (числовой ввод!) неотрицательное целое число и печатает новое число, которое получается из исходного путем вычеркивания всех цифр **3**.
В программе описать рекурсивную целочисленную функцию **`Delete3(N)`**, значением которой является число, полученное из целого неотрицательного `N` удалением в его десятичной записи всех цифр 3 (*например*, 14353 → 145), или число `0`, если в `N` только цифры `3` (*например*, 333 → 0).

“Простое число”. Написать программу, которая вводит (с помощью числового ввода) целое число N (N>**1)** и определяет, является ли оно простым. Для проверки числа следует описать булевскую функцию **`IsPrime(N)`**. Требования: эта функция не должна обращаться к глобальным переменным и использовать операторы цикла и перехода. *Подсказка*: решать по аналогии с решённой на семинаре задачей №6 (на подсчёт делителей числа N). *Требования*: 1) вспомнить, какой диапазон достаточно исследовать на предмет делителей; 2) в решении не использовать вещественной арифметики.

12.26 – решать по аналогии с разобранной на семинаре задачей **12.25**. В программе описать рекурсивную целочисленную функцию без параметров для считывания (с клавиатуры) формулы и вычисления её значения.

12.27 – решать согласно образцу, данному на семинаре.

12.28 - в разделе операторов программы должен быть только один оператор **`PRINT`**, где **`PRINT`** – вызов рекурсивной процедуры без параметров, которая считывает выражение в инфиксной записи и печатает его в постфиксной форме. В основной программе не должно быть описано каких-либо переменных. *Внимание*: синтаксис исходного выражения описан в условии с помощью БНФ (про это не забывать, это важно в решении).
Примеры работы программы (все эти тесты должны пройти!):

`((a+(b-c))*d)` → **`abc-+d*`**

`a` → **`a`**

`(a+b)` → **`ab+`**

`((a+b)-(c*(d+e)))` → **`ab+cde+*-`**

`(((a+b)+b)+c)+d)` →> **`ab+b+c+d+`**

См. далее

12.33 - реализовать рекурсивную процедуру **Move(n:integer;A,B,C:char)**, где **A** – исходный стержень, **B** – промежуточный (вспомогательный) стержень, **C** – целевой (конечный) стержень. Идея работы процедуры. 1 ЭТАП: пусть мы сумели переложить (**n-1**) колец с **A** на **B**, используя **C** как промежуточный (это возможно с учетом подсказки к задаче). 2 ЭТАП: после этого перекладываем самое большое кольцо с **A** на **C**. 3 ЭТАП: остается переложить (**n-1**) колец с **B** на **C**, используя **A** как промежуточный стержень. ВНИМАНИЕ: на 1 и 3 ЭТАПАХ (им соответствуют рекурсивные вызовы) по-разному задаются фактические параметры к процедуре **Move** (т.к. меняется назначение (смысл) стержней). Печать сообщения о переносе дисков производится на 2 ЭТАПЕ.

Блок-6. Машины (дополнительные) по теме «Рекурсия, часть 2»

“Факториал ?” (5 очков) Написать программу, которая вводит (с помощью числового ввода) целое число N (N>**0)** и определяет, является ли оно факториалом некоторого числа k>0 (т.е. представимо ли N в виде N=1*2*3*…*k ?). Для проверки числа следует описать целочисленную функцию **F(N)**, которая возвращает k, если N=k!, а иначе – возвращает значение -1. *Требования*: эта функция не должна обращаться к глобальным переменным и использовать операторы цикла и перехода. *Подсказка*: решать по аналогии с решённой на семинаре задачей №6 (на подсчёт делителей числа N). Примеры: F(1)=1, F(2)=2, F(3)= -1, F(6)=3, F(24)=4, F(90)= -1, F(120)=5 и т.п.

”Формула ?” (5 очков) (это вариация на тему решённой задачи **12.25),** формулировка этой задачи**:**
Во входном файле задан непустой текст, за которым следует точка. Проверить, является ли этот текст правильной записью “формулы” (см. определение формулы к задаче 12.25).
Рекомендуется описать булевскую функцию **`F`** без параметров, которая считывает из начала буфера ввода символы и пытается распознать в них правильную *формулу*. При первом синтаксически неверном символе – функция прекращает дальнейший ввод из буфера и возвращает ответ **`false`**. Если функции удается выделить из начала буфера верную формулу – возвращает ответ **`true`**. Эта функция запускается из основной программы, после чего анализируется её результат. Если **`false`** – все понятно, программа печатает слово “false”. Если **`true`** – то надо проверить (прочитать) символ вслед за найденной формулой; если это точка, то программа печатает слово “true”, иначе - “false” (например, для случаев: **23.** или **2+3.** или **(2+3)-1.** Должно быть напечатано слово “false”)