Практическое занятие «Ассемблер–6»
31 марта 2020 года

В этой практике вызов каждой процедуры следует иллюстрировать вызовом из главной программы (если процедура не является вспомогательной и явно не вызывается из других частей вашего кода). Если не оговорено иное, считается, что массив передаётся в процедуру следующим образом: в регистре `eax` — длина, в регистре `esi` — адрес памяти.

1. Напишите процедуру `totalGCD`, принимающую массив 32-битных чисел со знаком и находящую совокупный НОД всех этих чисел. Например, 28, −70, 154, 98 → 14. Результат должен возвращаться в регистре `eax`. Напишите вспомогательную процедуру, принимающую в регистрах `eax` и `ebx` два 32-битных числа со знаком (или без знака, как то будет удобнее для вашей программы) и возвращающую в регистре `eax` их НОД.

2. Напишите функцию `pow`, которая возводит 32-битное знаковое число, переданное в регистре `eax` в степень беззнакового 8-битного числа, переданного в регистре `bl` (считаем, что показатель степени не слишком велик, так что результат входит в 32-битный регистр). С использованием написанной процедуры для заданного 32-битного знакового числа a найдите сумму заданных степеней. Результат запишите в 4-байтную переменную `sumPow`. Степени заданы в байтовом массиве `powers`; длина этого массива задана в переменной/константе `powLen`. Например, для входных данных $a = 3$, `powers` $= [0, 2, 4]$ в переменной `sumPow` должен получиться результат $3^0 + 3^2 + 3^4 = 1 + 9 + 81 = 91$.

3. Напишите функцию `isPalindrome`, которая проверяет, что заданный массив 16-битных чисел без знака является палиндромом (одинаково читается от начала к концу и от конца к началу). С её использованием для заданного массива `ar` 16-битных чисел без знака найдите максимальный по длине подмассив-палиндром. Если таких несколько, найдите любой. Длина массива `ar` задана в переменной/константе `arLen`. (Разумно организовать двойной цикл: внешний — по длине палиндрома, внутренний — по возможному месту начала.) Результирующий палиндром скопируйте в достаточно большой буфер `bestPal`, длина его должна быть записана в 4-байтную беззнаковую переменную `bestLen`.

4. Напишите процедуру `computeValue`, принимающую массив 32-битных знаковых коэффициентов a_i ($i = 0, \ldots, n$) многочлена $p(x) = \sum a_i x^i$ (младшие элементы массива — старшие коэффициенты многочлена), 32-битную знаковую величину x^* в регистре `ebx` и вычисляющую значение $v = p(x^*)$ многочлена в данной точке, возвращая его в регистре `eax`. С использованием написанной процедуры найдите наибольшее значение многочлена в точках, заданных в массиве `xs`, содержащем 32-битные знаковые величины; длина этого массива задана в переменной/константе `xsLen`. Запишите найденный максимум в 4-байтную переменную `maxVal`.

   При вычислении значения многочлена используйте схему Горнера:

$$v_0 = 0,\ v_{i+1} = x^* \cdot v_i + a_{n-i}, \quad i = 0, \ldots, n, \quad v = v_n.$$

   Здесь индекс коэффициента не есть индекс в массиве!

5. Пара натуральных чисел называется *дружественной*, если второе из них равно сумме собственных делителей первого (то есть всех его делителей, исключая само число), а первое равно сумме собственных делителей второго. Например, дружественной парой являются числа 220 и 284:

$$284 = 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110, \quad 220 = 1 + 2 + 4 + 71 + 142.$$

Напишите программу, находящую все пары дружественных чисел (первое меньше второго), большее из которых не превосходит 100000, и записывающих и в достаточно большой массив `amicNums` парами друг за другом; длину полученного массива запишите в переменную `amicLen`. Напишите вспомогательную процедуру, принимающую в регистре `eax` 32-битное число без знака и возвращающую в том же регистре сумму его делителей.