Ткаченко Кирилл Станиславович
Инженер 1-й кат.
ФГАОУ ВО «Севастопольский государственный университет».

Эфиопское умножение

Предлагается программа для демонстрации так называемого «эфиопского умножения» на школьном алгоритмическом языке. Программа может быть полезна при изучении программирования, школьного алгоритмического языка и его конструкций, а также увлекающимся занимательной математикой. Приводится полный исходный текст и результаты работы программы.

На разных этапах развития математики существовали различные способы умножения целых положительных чисел. К этим способам относится и так называемое «эфиопское умножение», известное с глубокой древности [1, 2]. Известны программные реализации демонстрации эфиопского умножения на различных языках программирования [2].

Целью настоящей работы является разработка программы на школьном алгоритмическом языке, выполняющей демонстрацию эфиопского умножения.

Прежде чем перейти непосредственно к разработке этой программы, стоит заострить внимание на способе умножения, называемом эфиопским. Этот способ состоит в использовании операций сложения, удвоения, целочисленного деления на два и проверки четности числа. Поэтому операции удвоения, целочисленного деления на два и проверки четности числа будут реализованы отдельными алгоритмами.

Умножение выполняется в две колонки. Одна из них (в частности, левая) заполняется последовательно сверху вниз: множимое, затем результат целочисленного деления множимого на два, после — результат деления полученной величины и так далее, пока не получится в итоге единица.

Другая колонка (в частности, правая) заполняется последовательно сверху вниз: множитель, затем результат удвоения множителя, после — результат удвоения полученной величины и так далее, пока происходит заполнение левой колонки.

После этого вычеркиваются из правой колонки те значения, напротив которых в левой колонке находится четное число. Оставшиеся значения в правой колонке суммируются.

Например, при эфиопском умножении 17 на 34 происходят следующие действия:

17	34
8	~~68~~
4	~~136~~
2	~~272~~
1	544
	===
	578

А именно: вначале записываются множимое и множитель 17 и 34. Поскольку 17 — нечетное число, то 34 не вычеркивается:

17 34

17 делится нацело на два, 34 умножается на два. Получаются 8 и 68. Поскольку 8 — четное число, то 68 вычеркивается:

8 ~~68~~

8 делится нацело на два, 68 умножается на два. Получаются 4 и 136. Поскольку 4 — четное число, то 136 вычеркивается:

4 ~~136~~

4 делится нацело на два, 136 умножается на два. Получаются 2 и 272. Поскольку 2 — четное число, то 272 вычеркивается:

2 ~~272~~

2 делится нацело на два, 272 умножается на два. Получаются 1 и 544. Поскольку 1 — нечетное число, то 544 не вычеркивается:

1 544

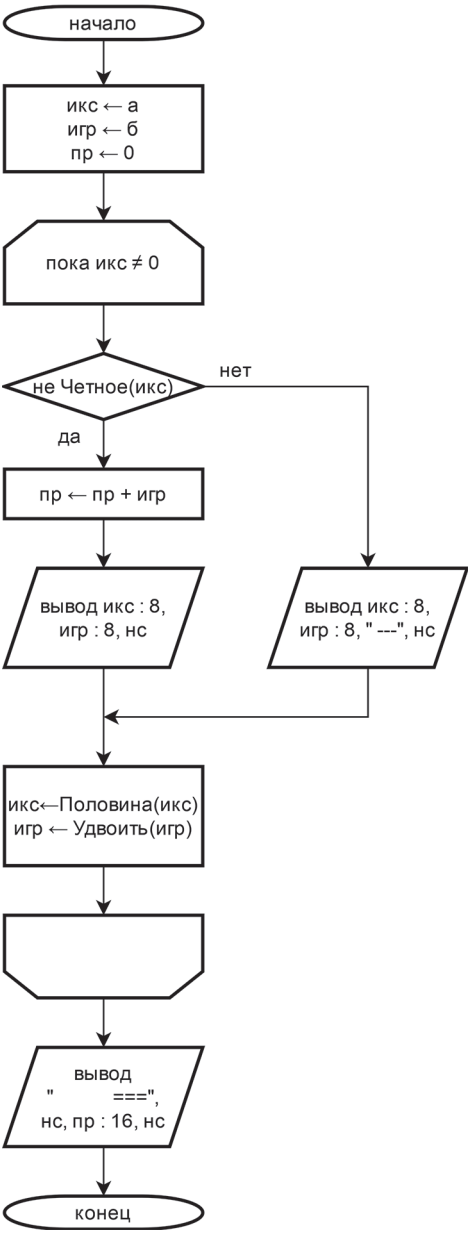

Рис. 1. Схема алгоритма

В левой колонке появилась единица, поэтому находится сумма невычеркнутых чисел в правой колонке, 34+544, которая и является результатом умножения: 578.

Пример процесса эфиопского умножения совпадает с [2].

Теперь можно приступить непосредственно к написанию программы. Программа начинает свое выполнение с алгоритма Потенциал, которые производит вызов алгоритма для демонстрации умножения 17 на 34:

```
алг Потенциал
нач
  ЭфиопскоеУмножение(17, 34);
кон
```

Следующий алгоритм, Эфиопское Умножение (цел а, цел б), производит непосредственную демонстрацию эфиопского умножения (схема алгоритма изображается на рисунке 1). В целочисленных аргументах а и б находятся множимое и множитель, соответственно, в целочисленных переменных икс, игр и пр запоминаются модифицируемые значения множимого (текущее значение в левой колонке), множителя (текущее значение в правой колонке) и накопленного результата произведения:

```
алг ЭфиопскоеУмножение(цел а, цел б)
нач
  цел икс, игр, пр;
```

Вначале инициализируются множимое, множитель и произведение:

```
  икс := а;
  игр := б;
  пр := 0;
```

Пока множимое не станет равным нулю:

```
  нц пока икс <> 0
```

Если текущее значение в левой колонке является нечетным, то к произведению добавляется текущее значение в правой колонке, и они выводятся:

```
    если не Четное(икс)
      то
        пр := пр + игр;
        вывод икс : 8, игр : 8, нс;
```

Иначе текущие значения для левой и правой колонок выводятся, при этом значение в правой колонке является вычеркнутым:

```
      иначе
        вывод икс : 8, игр : 8, " ---", нс;
    все
```

Следующее значение в левой колонке делится нацело на два, а в правой – удваивается:

```
    икс := Половина(икс);
    игр := Удвоить(игр);
  кц
```

Выводится найденное произведение:

```
  вывод "             ===", нс, пр : 16, нс;
кон
```

Алгоритм Удвоить (цел число) возвращает целое, которое является результатом удвоения целочисленного аргумента число:

```
алг цел Удвоить(цел число)
нач
  знач := число * 2;
кон
```

Алгоритм Половина (цел число) возвращает целое, которое является результатом целочисленного деления на два целого аргумента число:

```
алг цел Половина(цел число)
нач
  знач := div(число, 2);
кон
```

Алгоритм Четное (цел число)возвращает «да», если целочисленный аргумент число является четным числом, иначе возвращает «нет»:

```
алг лог Четное(цел число)
нач
  знач := mod(число, 2) = 0;
кон
```

Полный исходный текст программы находится в приложении А, результаты работы программы — в приложении Б, на снимке экрана приводится среда разработки с отработавшей программой.

Программа может быть полезна всем изучающим программирование, школьный алгоритмический язык и увлекающимся занимательной математикой, в том числе, школьникам.

Список литературы

1. Ancient Egyptian multiplication // Wikipedia. URL: https://en.wikipedia.org/wiki/Ancient_Egyptian_multiplication (датаобращения: 03.08.2020).

2. Ethiopian multiplication // Rosetta Code. URL: https://rosettacode.org/wiki/Ethiopian_multiplication (дата обращения: 03.08.2020).

Приложение А
Текст программы на школьном алгоритмическом языке

```
алг Потенциал
нач
  ЭфиопскоеУмножение(17, 34);
кон

алг ЭфиопскоеУмножение(цел а, цел б)
нач
  цел икс, игр, пр;
```

```
  икс := а;
  игр := б;
  пр := 0;

  нц пока икс <> 0
    если не Четное(икс)
      то
        пр := пр + игр;
        вывод икс : 8, игр : 8, нс;
      иначе
        вывод икс : 8, игр : 8, " ---", нс;
    все

    икс := Половина(икс);
    игр := Удвоить(игр);
  кц

  вывод "              ===", нс, пр : 16, нс;
кон

алг цел Удвоить(цел число)
нач
  знач := число * 2;
кон

алг цел Половина(цел число)
нач
  знач := div(число, 2);
кон

алг лог Четное(цел число)
нач
  знач := mod(число, 2) = 0;
кон
```

Приложение Б
Результаты работы программы

```
      17      34
       8      68 ---
       4     136 ---
       2     272 ---
       1     544
             ===
             578
```