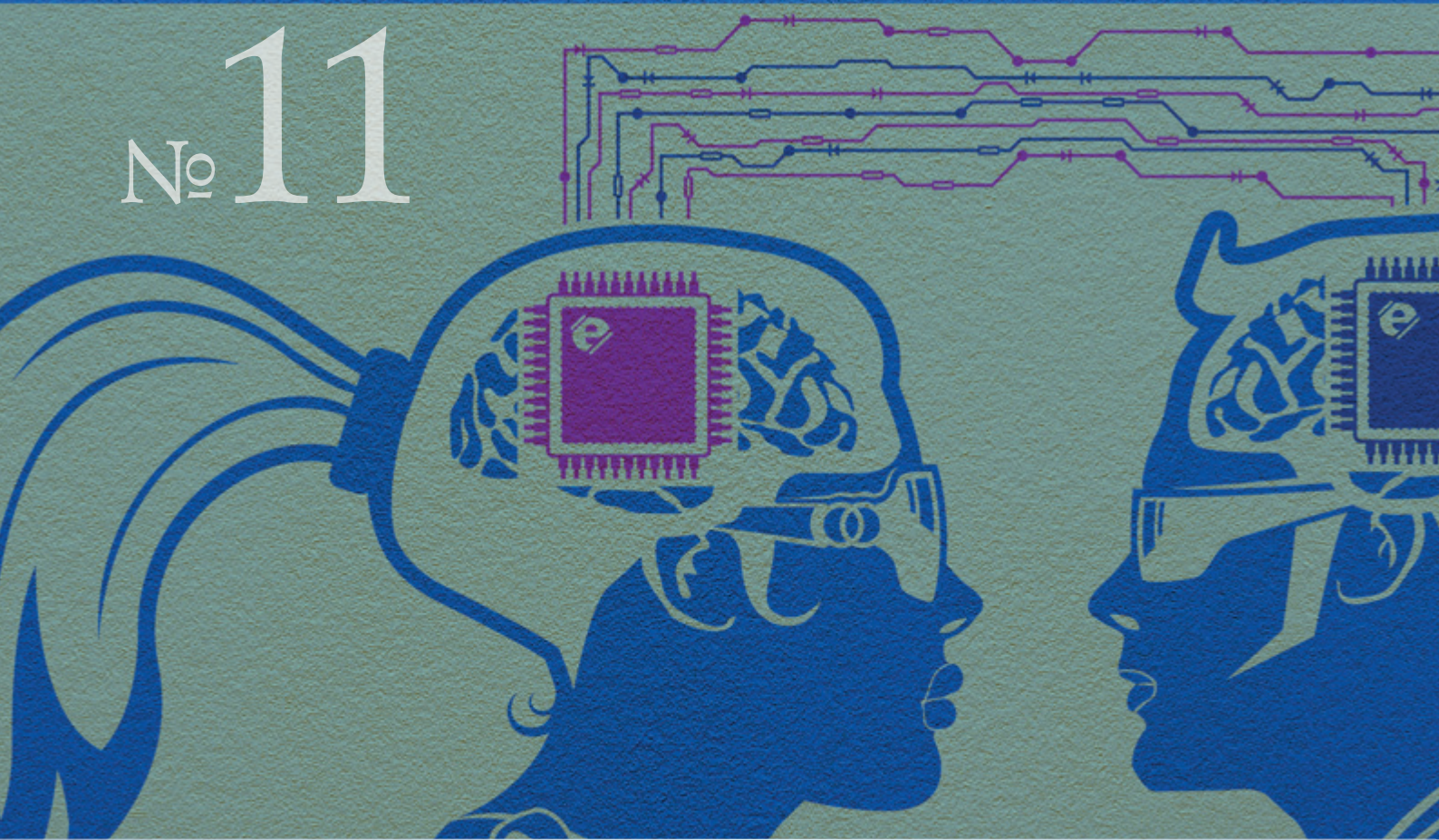
№ 11 1–15 июня 2011 Основана в 1995 г. inf.1september.ru
УЧЕБНО-МЕТОДИЧЕСКАЯ ГАЗЕТА ДЛЯ УЧИТЕЛЕЙ ИНФОРМАТИКИ

ИНФОРМАТИКА

/ тема номера:

ЭВМ-практикум

№ 11



издательский дом Первое сентября

1september.ru | ИНФОРМАТИКА Индексы подписки Почта России – 79006 (инд.); – 79574 (орг.) Роспечать – 32291 (инд.); 32591 (орг.)

О ТЕМЕ НОМЕРА

Удивительный (или нет?) факт: несмотря на объективно возрастающий уровень абстракции и в информатике вообще, и в школьной информатике, в частности (а уровень абстракции растет и на понятийном уровне, и на уровне используемых языков и средств программирования), продолжают разрабатываться учебные системы низкого машинного уровня. Не далее как в этом полугодии, в № 1/2011, наши читатели могли познакомиться с программой "Лампанель", разработанной К.Ю. Поляковым (программа была размещена на диске к № 2/2011). И вот новая (для наших читателей) разработка — "ЭВМ-практикум". "ЭВМ-практикум" является прямым сужением ассемблера процессоров x86 (это не плюс и не минус — просто констатация факта... хотя... скорее плюс). Для профильного курса эта среда может оказаться очень удобной и подходящей. Если бы еще, конечно, задачек и методики побольше... Ну, да нам не привыкать, сами настрогаем ☺.

В НОМЕРЕ

НА ДИСКЕ

Материалы к этому номеру будут размещены на диске, вложенном в № 12.
Внимание!
Это неудобство нам придется пережить в последний раз.
С августа диск будет вложен в каждый номер журнала.

ЭЛЕКТРОННЫЕ МАТЕРИАЛЫ:

- Авторская сборка "ЭВМ-практикума" — специально для "Информатики"
- Исходные файлы и презентации к статьям номера

ИНФОРМАТИКА

ПОДПИСНЫЕ ИНДЕКСЫ: по каталогу «Газеты. Журналы» агентства «Роспечать» – 32291 (инд); – 32591 (орг.); по каталогу «Почта России» – 79006 (инд); – 79574 (орг.)

http://inf.1september.ru

Учебно-методическая газета для учителей информатики
Основана в 1995 г.
Выходит два раза в месяц

РЕДАКЦИЯ:
гл. редактор С.Л. Островский
редакторы
Е.В. Андреева,
Д.М. Златопольский (редактор вкладки "В мир информатики")
Дизайн макета И.Е. Лукьянов
верстка Н.И. Пронская
корректор Е.Л. Володина
секретарь Н.П. Медведева
Фото: фотобанк Shutterstock
Газета распространяется по подписке
Цена свободная
Тираж 3000 экз.
Тел. редакции: (499) 249-48-96
E-mail: inf@1september.ru
http://inf.1september.ru

ИЗДАТЕЛЬСКИЙ ДОМ "ПЕРВОЕ СЕНТЯБРЯ"
Главный редактор: Артем Соловейчик (Генеральный директор)
Коммерческая деятельность: Константин Шмарковский (Финансовый директор)
Развитие, IT и координация проектов: Сергей Островский (Исполнительный директор)
Реклама и продвижение: Марк Сартан
Мультимедиа, конференции и техническое обеспечение: Павел Кузнецов
Производство: Станислав Савельев
Административно-хозяйственное обеспечение: Андрей Ушков
Дизайн: Иван Лукьянов, Андрей Балдин
Педагогический университет: Валерия Арсланьян (ректор)

ГАЗЕТЫ ИЗДАТЕЛЬСКОГО ДОМА
Первое сентября – Е.Бирюкова
Английский язык – А.Громушкина
Библиотека в школе – О.Громова
Биология – Н.Иванова
География – О.Коротова
Дошкольное образование – М.Аромштам
Здоровье детей – Н.Сёмина
Информатика – С.Островский
Искусство – М.Сартан
История – А.Савельев
Классное руководство и воспитание школьников – О.Леонтьева
Литература – С.Волков
Математика – Л.Рослова
Начальная школа – М.Соловейчик
Немецкий язык – М.Бузоева
Русский язык – Л.Гончар
Спорт в школе – О.Леонтьева
Управление школой – Я.Сартан
Физика – Н.Козлова
Французский язык – Г.Чесновицкая
Химия – О.Блохина
Школьный психолог – И.Вачков

УЧРЕДИТЕЛЬ:
ООО "ЧИСТЫЕ ПРУДЫ"

Зарегистрировано ПИ № 77-72230 от 12.04.2001 в Министерстве РФ по делам печати
Подписано в печать:
по графику 03.05.2011,
фактически 03.05.2011
Заказ №
Отпечатано в ОАО "Чеховский полиграфический комбинат"
ул. Полиграфистов, д. 1,
Московская область,
г. Чехов, 142300

АДРЕС ИЗДАТЕЛЯ:
ул. Киевская, д. 24,
Москва, 121165
Тел./факс: (499) 249-31-38

Отдел рекламы:
(499) 249-98-70
http://1september.ru

ИЗДАТЕЛЬСКАЯ ПОДПИСКА:
Телефон: (499) 249-47-58
E-mail: podpiska@1september.ru

Нас продают. Нас покупают

Итак, Microsoft купила Skype. За о-о-очень большие деньги. Бизнес, конечно, бизнесом, но сделки такого масштаба касаются не только самих компаний, но и миллионов (в данном случае — сотен миллионов) пользователей продаваемого/покупаемого продукта. Собственно, продают ведь не только (и не столько!) сам продукт — программный код. Сколь бы ни был он эксклюзивен, большие IT-корпорации уж точно могут при необходимости "сделать так же". Продают и покупают нас ☺. Именно мы — "клиентская база" — и стоим таких сумасшедших денег.

Так если продают и покупают нас, может быть, можно надеяться на то, что и нам что-то перепадет? Например, новые функциональные возможности продукта, ускорение выхода новых версий, еще какие-нибудь "вкусности"? Посмотрим. Сотни миллионов пользователей Skype практически затаили дыхание и ждут… как бы не стало хуже. Но что такого особенного нашли эти миллионы в Skype?

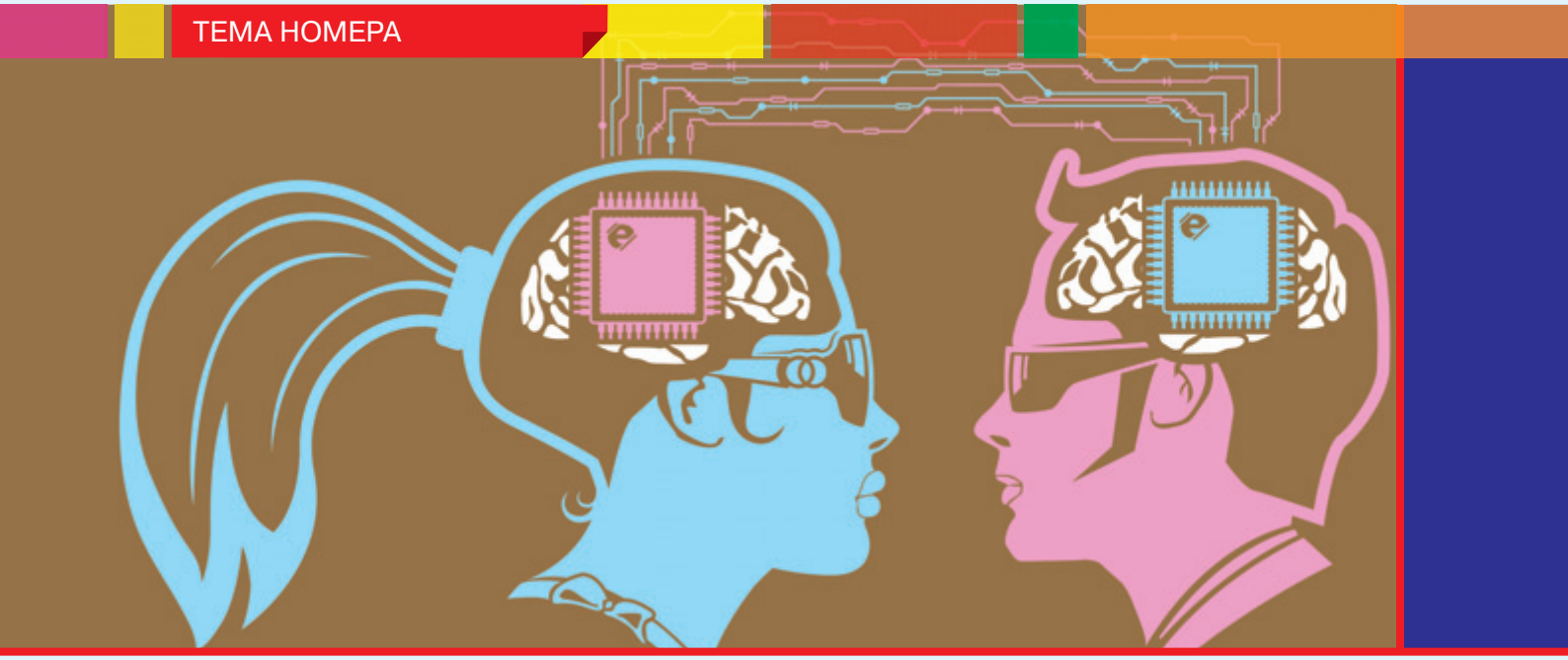
Компания Skype была основана двумя предпринимателями — шведом Никласом Зеннстрёмом и датчанином Янусом Фриисом. Авторами программного обеспечения стали эстонские программисты Ахти Хеинла, Приит Касесалу и Яан Таллинн, известные по созданию программы для файлообмена Kazaa. Первый релиз программы и сайт появились в сентябре 2003 года.

Skype создавалась в первую очередь для голосового и видеообщения. Skype использует P2P-архитектуру (Peer-to-Peer, равный-к-равному). Это позволяет сети легко масштабироваться до очень больших размеров, не "упираясь" в возможности центральных серверов. Фактически центральный сервер нужен только для установки связи. После того как связь установлена, компьютеры пересылают данные напрямую друг другу (если между ними есть прямая связь) или через Skype-посредник (суперноду). В частности, если два компьютера, находящиеся внутри одной локальной сети, установили между собой Skype-соединение, то связь с Интернетом можно прервать, и соединение будет продолжаться вплоть до его завершения пользователями или какого-либо сбоя связи внутри локальной сети.

Впрочем, необходимость в достаточном количестве супернод не так давно обернулась большими проблемами. В декабре 2010 г. Skype "лежал" почти двое суток. Проблема была вызвана ошибкой в самой программе. Вследствие этой ошибки после обновления клиенты Skype для Windows "падали". И те, которые были супернодами, тоже падали. В результате в сети не осталось достаточного количества супернод.

Skype очень нелюбим спецслужбами. Причина нелюбви в том, что трафик Skype зашифрован и не поддается декодированию на перехватывающем узле (все известные "прослушки" Skype слушают именно пользовательский компьютер, но не канал). Зато, помимо шифрования, кодеки Skype отлично сжимают трафик, что позволяет общаться голосом даже при достаточно узких каналах. Но в любом случае, как мы уже отмечали, Skype позволяет пользователям и с помощью текстовых сообщений устраивать групповые чаты, посылать смайлики, хранить историю. Кроме того, Skype предоставляет возможность обмена файлами без ограничения размера.

В общем, не хотелось бы от всего этого отказываться ☺. Но скорее всего и не придется. Ведь, напомним, покупают и продают прежде всего нас. И последнее, в чем должен быть заинтересован покупатель, — это отток пользователей. Поживем — увидим.



ЭВМ[1]-практикум

А.Г. Леонов,
Москва

В начале “славных дел” эры программирования и ЭВМ дела с алгоритмическими языками обстояли достаточно туго. Их просто не было. С ЭВМ приходилось “разговаривать” на ее языке. А язык ЭВМ — так называемые “коды процессора” — очень слабо походил на человеческий. Если провести параллель между человеком и ЭВМ, то процессорные коды у человека можно представить как команды отдельным мышцам сокращаться или расслабляться. Трудоемкость программирования в те времена можно оценить, попытавшись “разложить” шаг человека в виде последовательности команд различным мышцам. Тут очень легко ошибиться и разучиться ходить.

Кроме того, различные ЭВМ имели собственную систему команд, набор чисел, связанных с определенными действиями процессора ЭВМ.

```
 8 ******************
 9
10          ORG  $4000
11 A1       =    $3C
12 A2       =    $3E
13 A4       =    $42
14 AUXMOVE  =    $C311
15
16 ************************
17 * SETUP - move data for VTOC
18 * and catalog to auxmem at
19 * B000-B3FF (pseudo trk 11
20 * 0-3)
21 ************************
22 SETUP    LDA  #<VTOC
23          STA  A1
24          LDA  #>VTOC
25          STA  A1+1
26          LDA  #<END
27          STA  A2
28          LDA  #>END
29          STA  A2+1
30          LDA  #$B0
31          STA  A4
32          LDA  #$B0
33          STA  A4+1
34          SEC
35          JMP  AUXMOVE
36
37          DS   4
38 **************
```

Так же и у человека — каждый орган имеет собственные команды, и процесс пищеварения не сводится только к управлению различными мышцами.

Только в 1952 году американка Грейс Хоппер (см. фото на с. 5) создала первый в мире язык **Ассемблер**[2]. Он включал в себя символьную систему команд (а не набор числовых кодов), библиотеки процедур (набор готовых алгоритмов) и специальную программу для перевода текста программы в машинный код — **компилятор**[3].

Это был первый шаг в создании *языка программирования*, понятного и человеку, и компьютеру. В настоящее время языки программирования настолько далеко “ушли” от языка ЭВМ, что есть программисты, которые даже и не знают о существовании языков ЭВМ. Однако по-прежнему процессоры современных компьютеров имеют свои собственные языки, а небольшое практическое знакомство с ассемблером одной из ЭВМ позволит лучше понять, как “думает” ЭВМ, и научит составлять более эффективные алгоритмы.

[1] ЭВМ — Электронная Вычислительная Машина. Термин “компьютер” является синонимом аббревиатуры “ЭВМ”. После появления персональных компьютеров (англ. — *personal computer, PC*) термин “ЭВМ” практически вытеснен из употребления и заменен заимствованным термином “компьютер”.

[2] Название *ассемблер* произошло от английского слова *assemble* — собирать, компоновать.

[3] Перевод с ассемблера в машинные коды (язык ЭВМ) называется **компиляцией** (англ. *to compile* — составлять, собирать), а программа перевода — **компилятором**.

Эту задачу мы и попробуем решить с помощью **ЭВМ-практикума**[4].

ЭВМ-практикум — это программная система, в которой в качестве объектов выступают оперативная память и регистры центрального процессора некой виртуальной ЭВМ (модифицированный и сокращенный вариант Intel x86). С помощью ЭВМ-практикума можно не только непосредственно редактировать эти объекты, но и создавать программу на ассемблере этой ЭВМ, выполнять ее непрерывно или по шагам и наблюдать, как меняются объекты — память, регистры, стек и пр.

Грейс Хоппер

ЭВМ-практикум предназначен для изучения в течение 4–6 часов, при этом обучаемый научится составлять простейшие программы на ассемблере (в кодах) ЭВМ и тем самым освоит общие принципы работы и устройство процессоров современных ЭВМ. Список задач для подобных занятий можно найти в конце этой статьи, а также на сайте *http://e.kumir.su*.

На экране ЭВМ-практикума мы видим *регистры* процессора, программу, *стек*, *флаги состояния* и окно памяти.

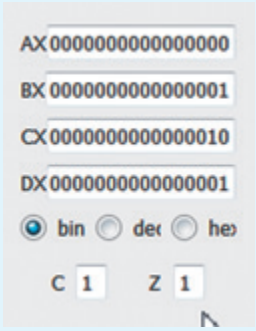
Регистры ЭВМ — это величины нашей программы. К сожалению, у нас все устроено почти “как в жизни”, поэтому величин у нас немного, всего 4. В каждой из них мы можем хранить целое число от 0 до 65 535 = $2^{16} - 1$. О таких одинаковых регистрах говорят, что это регистры *общего назначения* и что их *разрядность* 16 бит. Их имена — **AX**, **BX**, **CX**, **DX**.

У нас есть еще две скрытые от нас величины, которые в реальных процессорах доступны для программиста. Это два специальных регистра — **SP** (англ. *Stack Pointer* — указатель стека) и **IP** (англ. *Instruction Pointer* — счетчик команд). Первый (невидимо) указывает на вершину стека, а второй при пошаговом выполнении виден и указывает на текущую выполняемую команду — номер строки программы. Однако мы все-таки можем на них воздействовать при помощи некоторых команд нашей программы. Но об этом чуть позже.

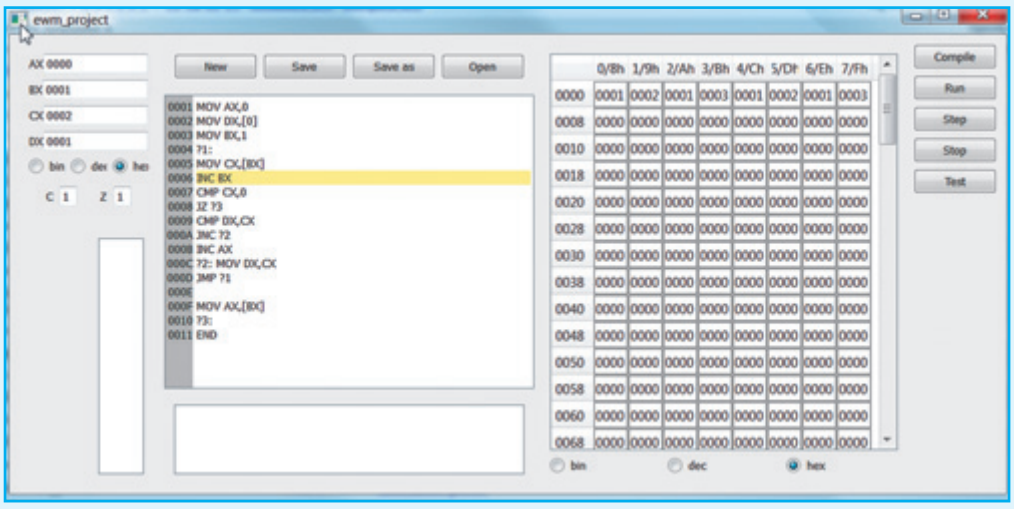
За результатами выполнения команд “следят” два битовых флага — C (англ. *Carry status* — флаг переноса) и Z (англ. *Zero status* — флаг нуля). Состояние последних отображается в виде битовой величины (0 или 1): флаг поднят (1) или опущен (0) — и зависит от результатов предыдущей команды. Из названия следует, что если результат выполненной команды равен 0, то флаг Z = 1, если результат отличен от 0, то Z = 0.

С флагом C немного сложнее. Зададимся вопросом, что будет, если результат операции “не поместится” в регистр. Так произойдет, если сумма больше 65 535 или разница меньше 0. В этом случае операции производятся “по модулю” 2^{16} (в регистре будет остаток) и символизирует такую несправедливость флаг C = 1.

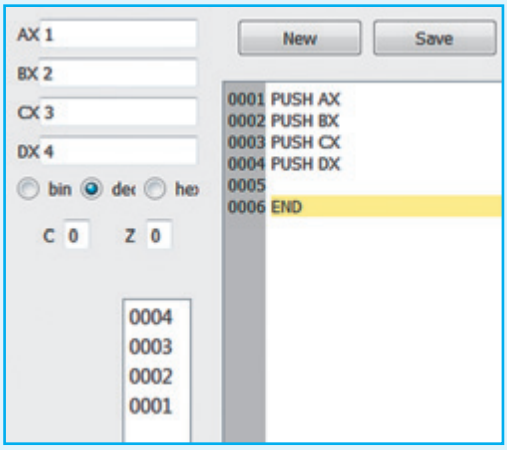
Содержимое регистров ЭВМ (AX-DX) мы можем видеть или в обычном для программистов шестнадцатеричном представлении, или в двоичном виде (особо любимом ЭВМ), и даже в привычном для нас, десятичном виде. Любое из этих представлений можно редактировать.



Флаги также можно редактировать, устанавливая значение в 0 или 1.



[4] ЭВМ-практикум распространяется свободно на условиях лицензии GNU 2.0. Данная лицензия разрешает вам или вашей организации бессрочно использовать ЭВМ-практикум на любом количестве компьютеров в любых целях без оформления каких-либо дополнительных документов. ЭВМ-практикум разработан совместно НИИСИ РАН и мехматом МГУ им. М.В. Ломоносова.



Значения SP и IP явно менять нельзя.

Окно стека[5] отображает содержимое стека в шестнадцатеричном представлении. Если SP = 0, то окно стека пустое. Если в стеке, например, 2 элемента (SP = 2), то они оба визуализируются, при этом вершина стека находится сверху. При этом в процессе выполнения программы всегда показывается окно стека так, чтобы содержимое вершины было изображено. Содержимое окна стека менять нельзя.

Как устроена память

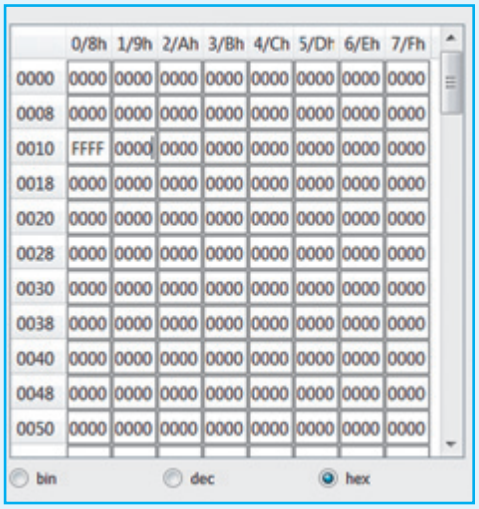
Память ЭВМ в целом — это таблица некоторых величин с индексом от 0 до некоторого максимального значения. Величины могут иметь различный размер, например, *бит*[6], *байт*, *слово* и т.д.

Бит может хранить один разряд, 0 или 1.То есть в бите можно закодировать два целых числа — 0 и 1.

Байт — группа из 8 бит: $i_7, i_6, i_5, i_4, i_3, i_2, i_1, i_0$. Байт может находиться в $2^8 = 256$ состояниях, их обозначают числами от 0 до 255 (десятичное) или от 00 до FF (шестнадцатеричное).

Слово еще больше — это два байта. Слово может принимать $2^{16} = 65\,536$ состояний. Биты в слове нумеруются с нуля и справа налево: $i_{15}i_{14}i_{13}i_{12}i_{11}i_{10}i_9i_8\,i_7i_6i_5i_4i_3i_2i_1i_0$. Одним словом представляются неотрицательные целые числа (Z_+) в диапазоне 0..65 535 ($x = i_{15} * 2^{15} + i_{14} * 2^{14} + \ldots + i_0$)[7]. Индекс слова в памяти называется его *адресом*.

Содержимое памяти (окно памяти) визуализируется в виде шестнадцатеричных слов (двухбайтовых кодов) или в двоичном, или в десятичном представлении. Общий объем памяти — 512 слов с адресами от 0 до 01FFh. Память отображается по 8 слов в строке. Строки нумеруются от 0 до 01F8h. Редактирование памяти можно выполнять непосредственно в окне памяти вне зависимости от его представления. Так, если в третью строку адреса 0010 введено число FFFF, то содержимое окна примет следующий вид.

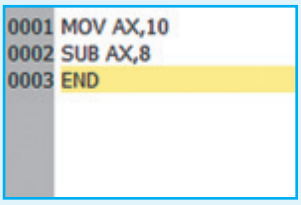


Как работает процессор

Процессор — это “мозг” ЭВМ, выполняющий все команды, а регистры, как мы знаем, — это внутренняя память процессора. В общих чертах работу процессора можно описать следующим основным циклом:

```
нц
. прочесть команду
  по адресу <арг IP>  в  <рез x>
. IP установить на следующую команду
. выполнить команду <арг x>
кц
```

Содержимое программы ЭВМ состоит построчно из номера строки (в шестнадцатеричном виде) и команды; если команда отсутствует (нет никакой операции), то поле команды пустое.



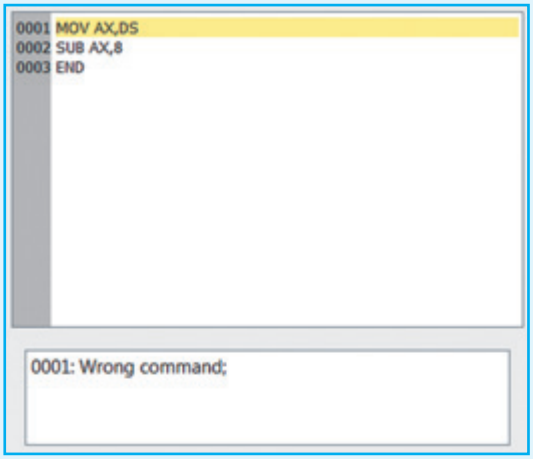
Программа начинается со строки с номером 0001. Последняя команда в программе — команда END. Эту команду нельзя удалить и нельзя “пройти” сквозь нее “ниже” программы.

При редактировании поля с командой синтаксическая правильность анализируется при выполнении программы или нажатии кнопки “компилировать”. Если строка синтаксически некорректна, то появляется предупредительное сообщение в строке сообщений, а строка выделяется цветом.

[5] **Стек** (англ. *stack* — стопка) — набор данных с доступом *LIFO* (англ. *Last In — First Out*, “последним пришел — первым вышел”). Чаще всего принцип работы стека сравнивают со стопкой тарелок (или с магазином автомата): чтобы взять вторую сверху, нужно сначала снять верхнюю.

[6] **Бит** (англ. ***bi**nary digit*) — один двоичный разряд в двоичной системе счисления, одна из самых маленьких единиц измерения информации. Перевод бита в состояние 0 принято называть *очисткой*, а в состояние 1 — *установкой* бита.

[7] Одним 16-битным словом также можно представить целые числа (Z) в диапазоне –32 768..32 767 ($x = i_{14} * 2^{14} + i_{13} * * 2^{13} + \ldots + i_0 - i_{15} * 2^{15}$). Знак числа Z определяется состоянием старшего бита: при $i_{15} = 0$ число больше либо равно 0; при $i_{15} = 1$ — меньше 0.



В ЭВМ-практикуме в программе можно копировать, вставлять и удалять строки. При выполнении команды END происходит возврат в режим редактирования. Кроме того, выполнение программы можно в любой момент прервать, далее можно продолжить выполнение непрерывно или по шагам. По завершении работы ЭВМ-практикума созданную программу можно сохранить в файле.

Фактически ЭВМ-практикум отличается от программирования в кодах только тем, что в нем можно вводить символические (ассемблерные) обозначения для команд, регистров и видов адресации, а также указывать в командах переходов не только абсолютные адреса памяти, но и числовые метки вида ?hhhh (где hhhh — шестнадцатеричная константа).

Пусть в некоторый момент IP = 2 для программы (процессор готов выполнить вторую строчку) (см. рис. слева).

После выполнения шага работы процессора в регистре BX окажется 6, а регистр IP = 3 (адрес следующей команды) (рис. справа).

Команды процессора

Каждая команда нашего процессора занимает одну строчку (в памяти команд) и может иметь до двух аргументов. Всего в нашем процессоре 27 команд, при помощи которых можно писать программы. Полный список команд и дополнительные материалы можно найти на сайте *http://e.kumir.su*. Здесь мы рассмотрим только некоторые из них (см. табл. ниже).

Приведем некоторые пояснения. В двухаргументных командах аргументы принято называть *получателем* (первый аргумент, d-destination) и *источником* (второй аргумент, s-source), если аргумент один, то он получатель (аргумент, d-destination).

Итак, d- и s-аргументы команды могут быть:

- регистром AX, BX, CX, DX (например, AX);
- содержимым памяти по адресу, содержащемуся в регистре (например, [BX]);
- содержимым памяти (например, [100]);
- *i*-й ячейкой массива с адресом начала массива в регистре (например, 2(CX), если CX = 100, то это то же самое, что и [102]).

Кроме того, s (2-й аргумент команды) может быть шестнадцатеричной константой, например, $20_{16} = 32$.

Обозначения в колонках битов Z и C говорят о следующем:

0 — в результате выполнения команды бит устанавливается в 0;

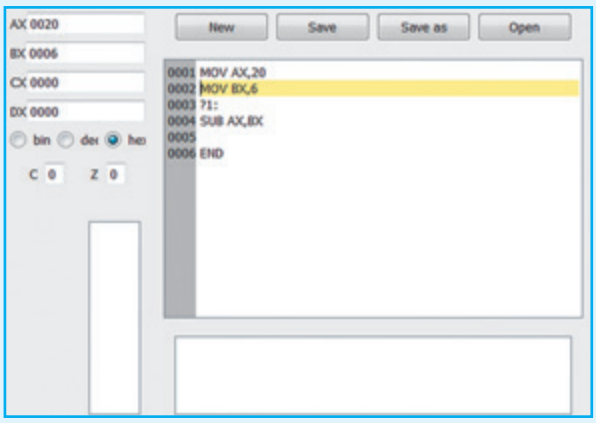
1 — в результате выполнения команды бит устанавливается в 1;

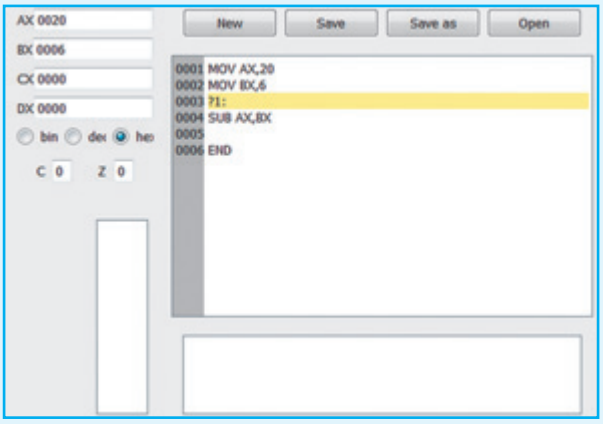
✓ — значение бита зависит от результата выполнения команды;

– — состояние бита в результате выполнения не изменяется.

Что касается команды сравнения CMP, то в результате операции значения ее аргументов не изменяются, а лишь сказываются на установке битов Z и C.

Команда	Действие	Флаг Z	Флаг C	Примеры
DEC d	d := d + 1	✓	-	DEC AX
INC d	d := d - 1	✓	-	INC [BX]
ADD d,s	d := d + s	✓	✓	ADD AX,20
SUB d,s	d := d - s	✓	✓	SUB [CX],1
ADC d,s	d := d + s + C	✓	✓	ADC 1(CX),2(CX)
SBB d,s	d := d – s - C	✓	✓	SBB DX,[DX]
MOV d,s	d := s	-	-	MOX AX,[100]
CMP d,s	d - s	✓	✓	CMP AX,[BX]





Рассмотрим простой пример.

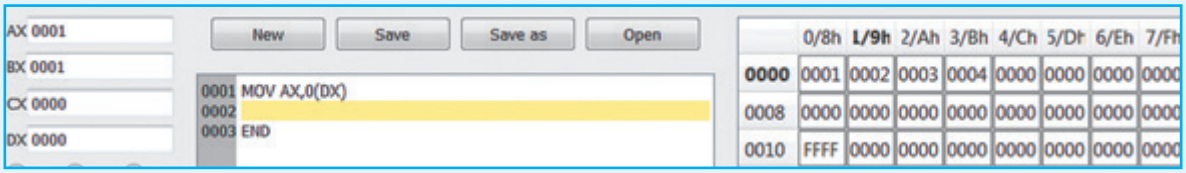
Пусть в регистре DX содержится адрес памяти, начиная с которого в ней расположены четыре числа. Найти их сумму (результат должен находиться в регистре AX), в предположении, что все числа не превышают 1000.

Решение

Так как все 4 числа не больше 1000, то сумма не больше 4000, а $4000 < 2^{16}$. То есть *переполнения* не будет.

Тогда для загрузки в AX первого числа можно воспользоваться командой

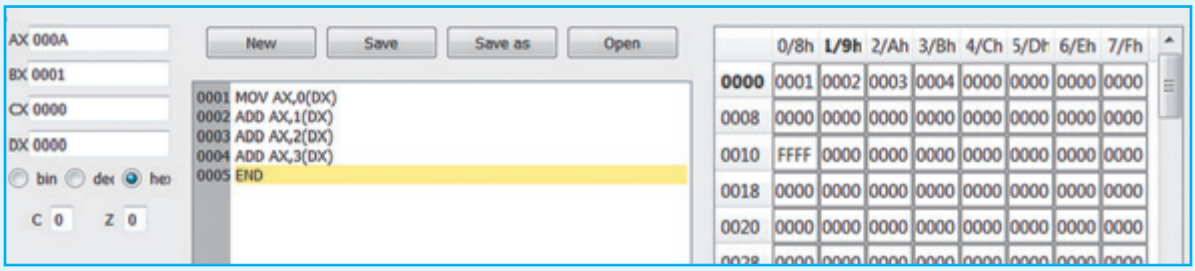
`MOV AX,[DX]` или `MOV AX,0(DX)`, которые приведут к одному и тому же результату:



Дальше останется сложить содержимое регистра со следующими тремя числами:

```
ADD AX,1(DX)
ADD AX,2(DX)
ADD AX,3(DX)
```

Пусть DX = 0, а по адресу 0, 1, 2, 3 находятся числа 1, 2, 3, 4. Выполнение программы должно дать значение 10 в регистре AX или $000A_{16}$:



Команды переходов

Без изменения порядка выполнения программы трудно (а подчас трудоемко) реализовать различные алгоритмы, требующие итерации или выбора. Для этого в языке ЭВМ есть команды условного и безусловного переходов.

Единственный результат выполнения команды JMP (команда безусловного перехода) состоит в том, что она помещает в IP свой аргумент. Следующие четыре *команды* перехода изменяют содержимое регистра IP при выполнении некоторых условий. Все эти условия являются утверждениями о состоянии битовых флагов Z и C и обычно связаны с результатом выполнения предыдущей команды.

Если условие перехода (заданное в коде операции) выполнено, то результат выполнения команды перехода можно записать в виде:

IP := адрес команды перехода.

Если же условие не выполнено, то IP станет равным адресу следующей команды.

Команда CALL — вызов процедуры — также осуществляет переход, но не только кладет свой аргумент в IP, но и "запоминает" точку возврата (адрес следующей команды) в стеке. Команда RET осуществляет возврат из процедуры, "вспоминая" из стека, куда нужно вернуться.

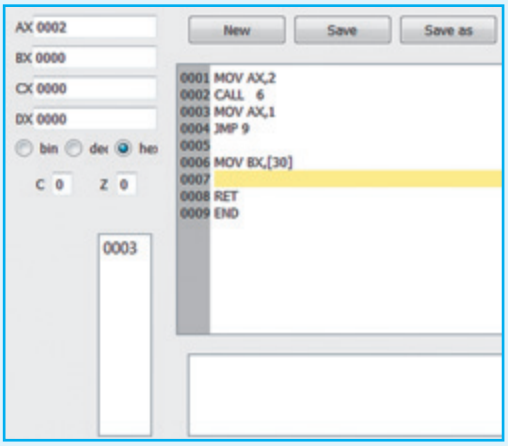
В следующем примере показано состояние регистров при вызове процедуры:

Строка	Программа	IP (после)	SP	Содержимое стека
0001	MOV AX,2	2	0	пусто
0002	CALL 6	6	1	3
0003	MOV AX,1	4	0	пусто
0004	JMP 9	9		пусто
0005				
0006	MOV BX,[30]	7	1	3
0007				
0008	RET	3	0	
0009	END			пусто

Выполнение этой программы будет происходить следующим образом:

1. IP = 2 — прочитана команда MOV.
2. IP = 6 — адрес подпрограммы из аргумента, [++SP] = 3 — адрес возврата.

3. Выполняется процедура с адресом (IP = 6).
4. ...
5. IP = [SP--], возврат из процедуры по команде RET.
6. IP = 3, далее будет выполняться программа.
7. ...



Ниже приведен неполный список команд перехода:

Команда	Действие	Флаг Z	Флаг C
JMP d	IP := d	-	-
JZ d	Если Z = 1 ⇨IP := d	-	-
JNZ d	Если Z = 0 ⇨IP := d	-	-
JC d	Если C = 1 ⇨IP := d	-	-
JNC d	Если C = 0 ⇨IP := d	-	-
CALL d	[++SP] := IP, IP := d	-	-
RET	IP := [SP--]	-	-

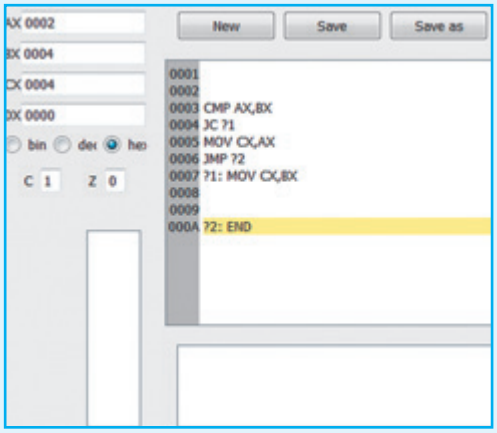
d — по-прежнему аргумент команды, но уже имеет другой смысл:
- номер строки перехода — шестнадцатеричное число (например, 100);
- метка (начинается со знака ?<число>), если такая есть (например, ?01).

Общий синтаксис команды Ассемблера имеет вид:
<необязательная метка><:> или
<необязательная метка><:> <команда> <аргументы>,
— где метка отделена двоеточием.

Примеры программ

Пример 1. CX := max(AX, BX). Максимум из двух чисел. Алгоритм достаточно очевиден.

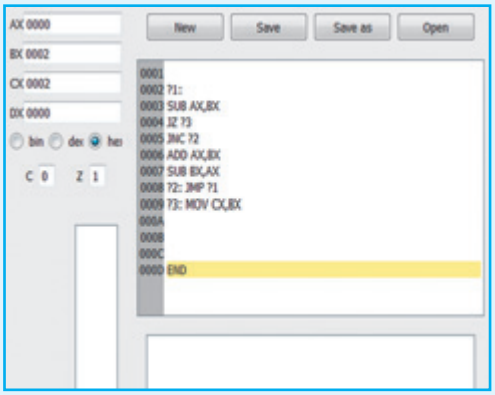
Запишем эту программу на языке Ассемблера и на алгоритмическом языке.



Ассемблер	Алгоритмический язык
	. . **алг цел** Максимум (**арг цел** AX,BX)
	. . **нач цел** CX
CMP AX,BX	. . . **если** AX > BX
JC ?1	. . . . .
MOV CX,AX	. . . . **то** CX := AX
JMP ?2	. . . . .
?1: MOV CX,BX	. . . . **иначе** CX := BX
	. . . **все**
	. . . **знач** := CX
?2: END	. . **кон**

Пример 2. CX := НОД(AX, BX), в предположении, что AX > 0 и BX > 0.

	. . **алг цел** НОД (**цел** M,N)
	. . **нач цел** CX,AX,BX
	. . . AX := M; BX := N
?1:CMP AX,BX	. . . **нц пока** AX <> BX
JZ ?3	. . . .
SUB AX,BX	. . . . AX := AX - BX
JNC ?2	. . . . **если** AX < **0**
	. . . . .
ADD AX,BX	. . . . . **то** AX := AX + BX
SUB BX,AX	. . . . . BX := BX - AX
	. . . . **все**
?2: JMP ?1	. . . **кц**
?3: MOV CX,BX	. . . CX := BX
	. . . **знач** := CX
END	. . **кон**



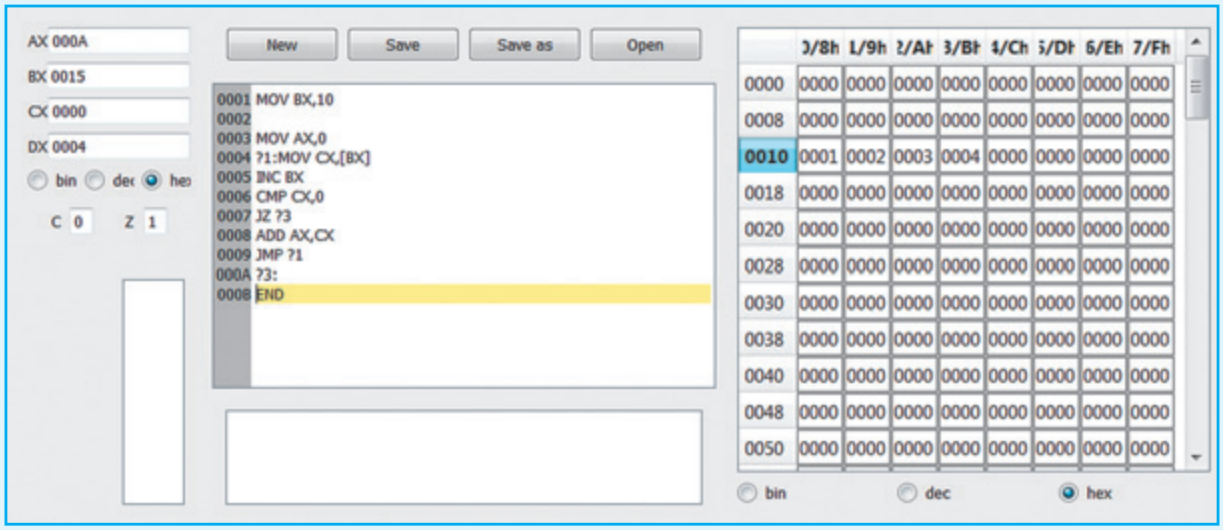
Известный алгоритм Евклида слегка нами модифицирован (алгоритмический язык) для удобства его перекодирования на Ассемблер. В первоисточнике сначала сравниваются величины AX и BX, и если AX > BX, то BX вычитается из AX, иначе AX вычитается из BX.

В скриншоте ЭВМ-практикума алгоритм еще более сокращен. Для AX = 8 и BX = 6 ответ CX = 2.

Пример 3. Подсчитать сумму в регистре AX элементов последовательности, расположенных начиная с адреса 10, при условии, что все элементы больше 0 и сумма не превышает $2^{16} - 1$. Последовательность оканчивается 0.

Решение. В алгоритмическом языке для чтения последовательности используются файлы, где эта последовательность и хранится. В нашем Ассемблере никаких файлов нет, поэтому хотя приведенные алгоритмы и похожи, но в них есть и явные различия. Так, чтение последовательности в алгоритмическом языке заканчивается, когда прочтен весь файл, а в нашем примере на Ассемблере признак конца — это 0, находящийся по адресу [BX]:

```
                    | . . использовать Файлы
                    | . . алг
                    | . . нач цел Ключ
MOV BX,10           | . . . открыть на чтение("p.txt", Ключ)
                    | . . . вывод "Сумма= ",Сумма (Ключ)
                    | . . . закрыть(Ключ)
                    | . . кон
                    | . . алг цел Сумма(арг цел Ключ)
                    | . . . дано | Последовательность целых чисел
                    | . . | в открытом файле с кодом Ключ
                    | . . . надо | Результат = сумма чисел
                    | . . | последовательности;
                    | . . нач цел CX, AX
MOV AX,0            | . . . AX := 0
?1:MOV CX,[BX]      | . . . нц пока не конец файла(Ключ)
INC BX              | . . . . ф_ввод Ключ,CX
CMP CX,0            | . . . .
JZ ?3               | . . . .
ADD AX,CX           | . . . . AX := AX + CX
JMP ?1              | . . . кц
?3:                 | . . . знач := AX
END                 | . . кон
```

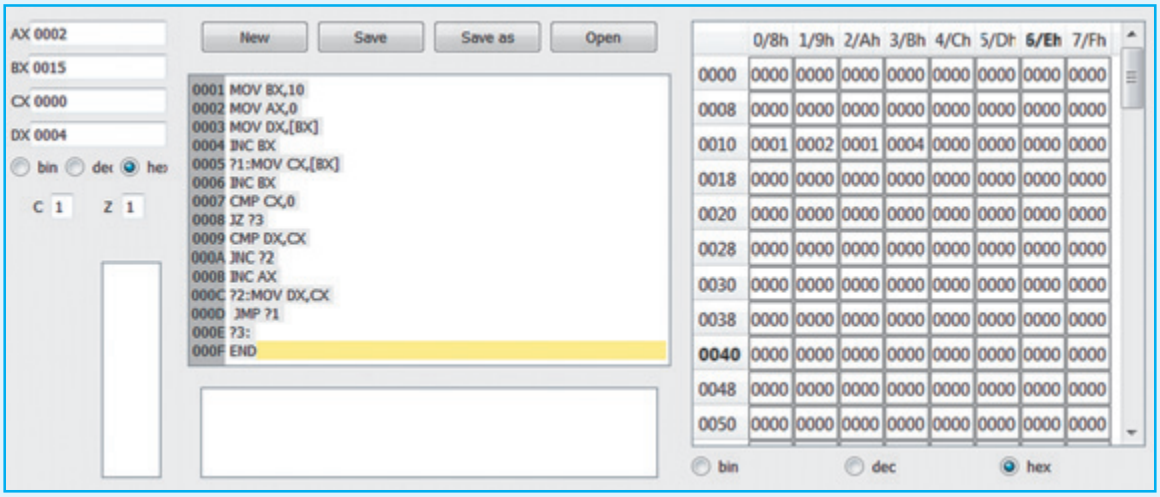


Для последовательности 1, 2, 3, 4 сумма будет равна 10, то есть AX = $000A_{16}$.

Пример 4. Подсчитать число возрастаний непустой последовательности в регистре AX. Последовательность расположена начиная с адреса 10, при условии, что все элементы больше 0. Последовательность оканчивается 0. Для одноэлементной последовательности ответ должен быть 0.

Решение. Эта задача похожа на предыдущую. Однако тут есть и собственная алгоритмическая сложность. Зная число элементов некоторой последовательности, больших предыдущего, невозможно определить, как изменится это число при добавлении к последовательности еще одного элемента, — необходимо также знать последний элемент последовательности. Последний элемент, в свою очередь, определен только на непустых последовательностях, что дано нам по условию. В процессе вычисления будем запоминать последний прочитанный элемент в DX, а также число элементов среди прочитанных, больших предыдущего в AX.

Для последовательности 1, 2, 1, 4 ответ AX = 2 (см. скриншот на с. 11).



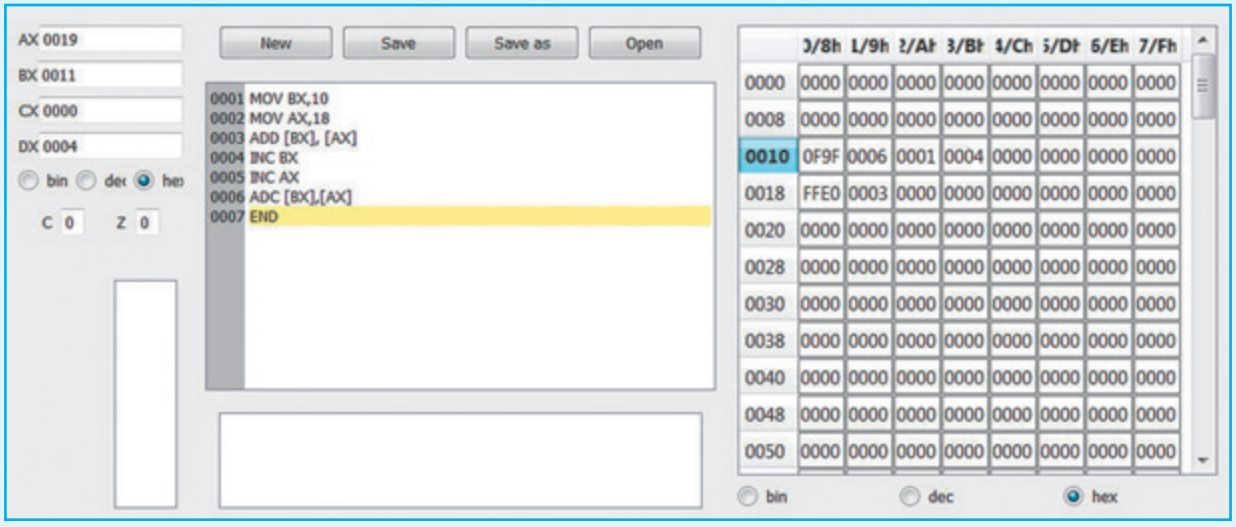

```
                        . . использовать Файлы
                        . . алг
                        . . нач цел Ключ
MOV BX,10               . . . открыть на чтение("p.txt", Ключ)
                        . . . вывод "Число возрастаний= ",Число возрастаний (Ключ)
                        . . . закрыть(Ключ)
                        . . кон
                        . . алг цел Число возрастаний (арг цел Ключ)
                        . . . дано | Последовательность целых чисел
                        . . | в открытом файле с кодом Ключ
                        . . . надо | Результат = число возрастаний
                        . . | непустой последовательности;
MOV AX,0                . . нач цел CX, AX, DX
MOV DX,[BX]             . . . AX := 0
INC BX                  . . . ф_ввод Ключ,DX
?1:MOV CX,[BX]          . . . нц пока не конец файла(Ключ)
INC BX                  . . . . ф_ввод Ключ,CX
CMP CX,0                . . . .
JZ ?3                   . . . .
CMP DX,CX               . . . . если DX < CX
JNC ?2                  . . . . . то AX := AX + 1
INC AX                  . . . . все
?2:MOV DX,CX            . . . . DX := CX
JMP ?1                  . . . кц
?3:                     . . . знач := AX
END                     . . кон
```

Примечание. Сложение длинных целых.

Пусть необходимо оперировать с целыми числами, выходящими за представимый с помощью одного слова диапазон ($>2^{16}$). Будем хранить число x в паре слов с адресами A и $A + 1$ и использовать 32 бита этой пары слов для представления x в двоичной системе счисления (старшие 16 бит — в слове $A + 1$, младшие — в слове A). Приведем пример кода, который добавляет длинное число, адрес которого записан в регистре AX, к длинному числу, адрес которого записан в регистре BX:

```
ADD [BX], [AX]    сложение младших слов чисел
INC BX            не изменяет флаг C
INC AX            не изменяет флаг C
ADC [BX],[AX]     учет переноса в старшие разряды и сложение старших слов чисел
```

Здесь мы используем флаг C. Пусть надо сложить два числа 20FBF и 3FFE0. Понятно, что при сложении младших частей мы получим переполнение и можем потерять разряд, однако он сохранен во флаге C. И операция ADC (сложение с C-флагом) корректирует операцию, результат равен 60F9F.



В заключение можно добавить, что в ЭВМ-практикуме реализована система тестирования аналогично системе тестирования в КуМире. Преподаватель может подготовить задачи с тестами для учеников, что позволит им самостоятельно проверять свои алгоритмы.

Задачи и упражнения

Упражнение 1. Напишите программу, которая помещает в регистр AX
a) абсолютную величину числа, лежащего в DX;
b) максимум из чисел, лежащих в AX, BX, CX;
c) сумма $2K + (2K - 2) + \ldots + 4 + 2$, где K — это содержимое BX.

В упражнениях 2–3 считается, что в памяти, по адресу 0 находится число, обозначающее число элементов последовательности, расположенных, начиная с адреса 1.

Упражнение 2. Напишите программу, после выполнения которой в регистре AX будет содержаться:
a) сумма модулей чисел;
b) адрес первого минимального числа;
c) число локальных максимумов последовательности.

Упражнение 3. Напишите программу, после выполнения которой
a) все числа увеличатся на 1;
b) числа образуют арифметическую прогрессию с начальным членом 1 и разностью 2;
c) в памяти будут записаны первые числа Фибоначчи $< 2^{16}$.

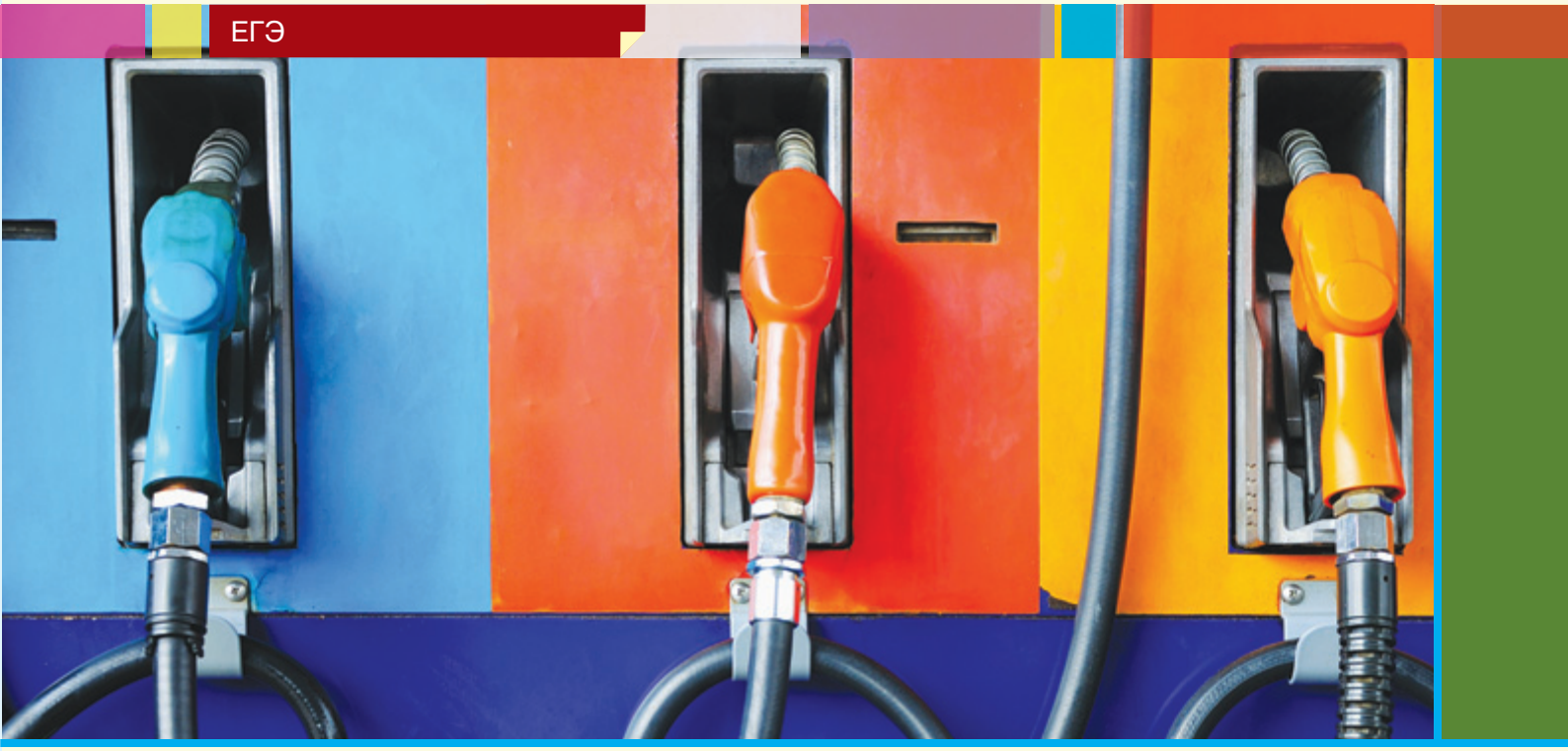
Задача 4. Целые числа из диапазона от 0 до $2^{128} - 1$ представляются в виде восьми последовательных слов памяти каждое. Напишите подпрограммы, реализующие для таких чисел операции:
a) сложение;
b) умножение на 2;
c) n-й член последовательности Фибоначчи.

Задача 5. Найти число минимальных элементов непустой последовательности за один просмотр, результат должен оказаться в регистре AX. Последовательность расположена начиная с адреса 10, при условии, что все элементы больше 0. Последовательность оканчивается 0.

Для одноэлементной последовательности ответ должен быть 1.

Задача 6. Найти второй по величине минимальный элемент непустой последовательности, результат должен оказаться в регистре AX. В случае отсутствия такого элемента ответ должен быть 0. Для одноэлементной последовательности ответ также должен быть 0.

Последовательность расположена начиная с адреса 10, при условии, что все элементы больше 0. Последовательность оканчивается 0.



Вспоминая прошлое, или Задача С4 из демонстрационного варианта ЕГЭ по информатике и ИКТ 2010 года

Д.М. Златопольский,
Москва

В статье описана методика решения задачи С4 из демонстрационного варианта Единого государственного экзамена по информатике и ИКТ 2010 года.

Напомним условие.

На автозаправочных станциях (АЗС) продается бензин с маркировкой 92, 95 и 98. В городе N был проведен мониторинг цены бензина на различных АЗС.

Напишите эффективную по времени работы и по используемой памяти программу (укажите используемую версию языка программирования, например, Borland Pascal 7.0), которая будет определять для каждого вида бензина, сколько АЗС продают его дешевле всего. На вход программе в первой строке подается число данных о стоимости бензина. В каждой из последующих N строк находится информация в следующем формате:

<Компания> <Улица> <Марка> <Цена>

— где <Компания> – строка, состоящая не более чем из 20 символов без пробелов, <Улица> – строка, состоящая не более чем из 20 символов без пробелов, <Марка> – одно из чисел – 92, 95 или 98, <Цена> – целое число в диапазоне от 1000 до 3000, обозначающее стоимость одного литра бензина в копейках. <Компания> и <Улица>, <Улица> и <Марка>, а также <Марка> и <Цена> разделены ровно одним пробелом. Пример входной строки:

Синойл Цветочная 95 2250

Программа должна выводить через пробел 3 числа – количество АЗС, продающих дешевле всего 92-й, 95-й и 98-й бензин соответственно. Если бензин какой-то марки нигде не продавался, то следует вывести 0.

Пример выходных данных:

12 1 0

В целом можно сказать, что обсуждаемая задача представляет собой задачу нахождения количества минимальных значений цены для каждой из трех марок бензина.

Основные положения методики решения задачи:

1. Информацию об очередной марке бензина на той или иной АЗС необходимо сохранить в величине строкового типа, не записывая ее в массив.

2. Из этой величины следует выделить марку бензина и ее цену на данной АЗС.

3. Полученную цену надо сравнить с минимальной ценой на соответствующую марку бензина для рассмотренных ранее АЗС.

Сначала рассмотрим вариант решения задачи без применения массивов.

В программе на школьном алгоритмическом языке используем следующие основные величины:

— *N* — число данных о стоимости бензина (см. условие задачи);

— *строка* — величина строкового типа, о которой шла речь в п. 1 чуть выше;

— *марка* — марка бензина (величина целого типа, равная 92, 95 или 98);

— *цена* — его цена (величина целого типа со значениями от 1000 до 3000);

— *мин*92, *мин*95, *мин*98 — минимальная цена, соответственно, на 92-й, 95-й и 98-й бензин;

— *кол*92, *кол*95, кол98 — искомые значения (количество АЗС, продающих дешевле всего 92-й, 95-й и 98-й бензин соответственно).

Обсудим методику выполнения пунктов 2 и 3.

| *Пункт 2*

Структуру строки с информацией об очередной марке бензина на той или иной АЗС можно представить в виде:

С	и	…	л		Ц	в	е	…	я		9	5		2	4	1	0
Название компании				*П*	*Название улицы*					*П*	*Марка*		*П*	*Цена*			

где *П* — пробел.

Поэтому для выделения марки бензина и ее цены на данной АЗС следует:

1) пропустить название компании:

```
номер_симв := 1 |Номер очередного
                |символа строки
нц
  |Переходим к следующему символу
  номер_симв := номер_симв + 1
  |пока не встретится 1-й пробел
кц_при строка[номер_симв] = " "
утв |Название компании пропущено
утв |Величина номер_симв указывает
    |на пробел
```

2) пропустить название улицы:

```
нц
  |Переходим к следующему символу
  номер_симв := номер_симв + 1
  |пока не встретится 2-й пробел
кц_при строка[номер_симв] = " "
утв |Название улицы пропущено
утв |Величина номер_симв указывает
    |на пробел
```

3) пропустить второй пробел (перед маркой бензина):

```
номер_симв := номер_симв + 1
```

4) “вырезать” два очередных символа марки и преобразовать их в число:

```
марка := лит_в_цел(строка[номер_симв :
номер_симв + 1], успех)
```

5) пропустить два очередных символа марки и пробел после нее:

```
номер_симв := номер_симв + 3
```

6) “вырезать” четыре очередных символа цены бензина и преобразовать их в число:

```
цена := лит_в_цел(строка[номер_симв :
номер_симв + 3], успех)
```

| *Пункт 3*

На этом этапе, зная значения величин *марка* и *цена*, нужно для каждой марки бензина выполнить действия, которые, например, для бензина марки 95 описываются так:

```
|Сравниваем значения цена и мин95
если цена < мин95
 то |Встретился новый минимум цены.
   |Запоминаем его
   мин95 := цена
   |Пока он - единственный
   кол95 := 1
 иначе
   |Проверяем, не является ли
   |значение цена
   |минимальным для данной марки бензина
   если цена = мин95
     то |Встретился еще один
        |минимум цены
        |на данную марку бензина.
        |Учитываем это
        кол95 := кол95 + 1
   все
все
```

Вся программа решения задачи имеет вид:

```
алг Задача_С4_2010
нач цел N, марка, цена, мин92, мин95,
        мин98, кол92, кол95, кол98,
        номер_симв, i,
    лит строка, лог успех
 |Начальные значения величин
 |кол92, кол95, кол98
  кол92 := 0; кол95 := 0; кол98 := 0
 |Начальные значения величин
 |мин92, мин95, мин98
 мин92 := 3001; мин95 := 3001;
```

```
мин98 := 3001
|Ввод значения N
вывод нс, "Введите число данных N "
ввод N
|Ввод и обработка каждой из строк
нц для i от 1 до N
 вывод нс, "Введите информацию о
           ",i,"-й АЗС "
 ввод строка
 |Обрабатываем очередную строку
 |1. Пропускаем название компании
 номер_симв := 1
 нц
  номер_симв := номер_симв + 1
 кц_при строка[номер_симв] = " "
 |2. Пропускаем название улицы
 нц
  номер_симв := номер_симв + 1
 кц_при строка[номер_симв] = " "
 |3. Пропускаем пробел перед
 |маркой бензина
 номер_симв := номер_симв + 1
 |4. Определяем марку бензина
 марка := лит_в_цел(строка[номер_симв :
          номер_симв + 1], успех)
 |5. Пропускаем марку бензина
 |и пробел после нее
 номер_симв := номер_симв + 3
 |6. Определяем цену бензина
 цена := лит_в_цел(строка[номер_симв :
         номер_симв + 3], успех)
 |7. Сравниваем полученную цену
 |с минимальной ценой
 |на соответствующую марку бензина
 выбор
  при марка = 92:
   если цена < мин92
    то
     мин92 := цена
     кол92 := 1
    иначе
     если цена = мин92
      то
       кол92 := кол92 + 1
     все
   все
  при марка = 95:
  …
  при марка = 98:
  …
 все
кц
|Выводим ответ
вывод нс, кол92, кол95, кол98
|Если бензина какой-то марки не было,
|соответствующее значение равно нулю
кон
```

Соответствующие программы на языках Бейсик и Паскаль (для всех вариантов решения) представлены на диске, прилагаемом к “Информатике” № 12/2011.

Можно также вместо шести величин *мин*92, *мин*95, *мин*98, *кол*92, *кол*95 и *кол*98 использовать два массива с индексами от 92 до 98:

1) с именем *мин* — для хранения минимальных цен на бензин разной марки;

2) с именем *кол* — для хранения искомых значений количества АЗС, продающих дешевле всего 92-й, 95-й и 98-й бензин соответственно.

При их использовании изменения в программе коснутся:

1) описания переменных величин;

2) задания начальных значений элементов массивов:

```
нц для i от 92 до 98
 кол[i] := 0
 мин[i] := 3001
кц
```

3) сравнения выделенной в каждой строке цены с минимальной ценой на соответствующую марку бензина для рассмотренных ранее АЗС:

```
выбор
 при марка = 92:
  если цена < мин[92]
   то
    мин[92] := цена
    кол[92] := 1
   иначе
    если цена = мин[92]
     то
      кол[92] := кол[92] + 1
    все
  все
 при марка = 95:
  если цена < мин[95]
   то
    мин[95] := цена
    кол[95] := 1
   иначе
    если цена = мин[95]
     то
      кол[95] := кол[95] + 1
    все
  все
 при марка = 98:
 …
 все
все
```

4) вывода ответа:

```
вывод нс, кол[92], кол[95], кол[98]
```

Размер программы существенно сокращается, если не использовать оператор варианта (выбора), а сравнивать значение *цена* со значением соответствующего элемента массива *мин* (с индексом *марка*):

```
|Сравниваем полученную цену
|с минимальной ценой
```

```
|на соответствующую марку бензина:
если цена < мин[марка]
 то
  мин[марка] := цена
  кол[марка] := 1
 иначе
  если цена = мин[марка]
   то
    кол[марка] := кол[марка] + 1
  все
все
…
```

Небольшую экономию размера программы дает использование индексов от 2 до 8 вместо индексов от 92 до 98. В этом случае в марке бензина следует выделять только второй символ-цифру. При этом программа на языке Паскаль может быть дополнительно упрощена за счет того, что значениями индексов элементов массивов *kol* и *min* в этом языке могут быть данные символьного типа:

```
var kol, min: array ['2'..'8']
    of integer;
```

Это позволит отказаться от преобразования символов “2”, “3”, …, “8” в число.

Но более существенное улучшение программы на языке Паскаль дает то обстоятельство, что пропустить название компании и название улицы можно используя оператор read с параметром символьного типа:

```
repeat
 read(c);
until c = ' ';
{Пропущено название компании}
repeat
 read(c);
until c = ' '; {Пропущено название улицы}
```

а получить числовые значения марки бензина и ее цены на данной АЗС можно, применив после этого оператор readln(marka, zena):

```
for i := 1 to N do
 begin
  repeat
   read(c);
  until c = ' ';
  repeat
   read(c);
  until c = ' ';
  readln(marka, zena);
  if zena < min[marka] then …
```

В заключение заметим следующее. Если обработка информации об очередной марке бензина на той или иной АЗС проводится после считывания всей строки (а в программах на школьном алгоритмическом языке и на языке Бейсик можно сделать только так), то получить значение цены и марки бензина можно без использования цикла — ведь нам известно “местоположение” соответствующих значений в каждой заданной строке[1]:

```
|1. Определяем цену бензина,
|"вырезая" и преобразовывая
|4 последних символа строки
цена := лит_в_цел(строка[длин(строка) -
        3 : длин(строка)], успех)
|2. Определяем марку бензина,
|"вырезая" и преобразовывая
|2 соответствующих символа строки
марка := лит_в_цел(строка[длин(строка)
         - 6 : длин(строка) - 5], успех)
|3. Сравниваем полученную цену с мини-
мальной ценой
|на соответствующую марку бензина
```

В заключение приведем программу решения рассмотренной задачи из домонстрационного варианта ЕГЭ.

```
var
min, ans: array[92..98] of integer;
c: char;
i, k, N, b: integer;
begin
  for i := 92 to 98 do
  begin
    min[i] := 3001;
    {допустимо и другое число,
     большее 3000}
    ans[i] := 0;
  end;
  readln(N);
  for i := 1 to N do
  begin
    repeat
      read(c);
    until c = ' '; {считана компания}
    repeat
      read(c);
    until c = ' '; {считана улица}
    readln(k,b);
    if min[k] > b then
    begin
      min[k] := b;
      ans[k] := 1
    end else
    if min[k] = b then
       ans[k] := ans[k] + 1;
  end;
  {если бензина какой-то марки не было,
   ans[i] осталось равным 0}
  writeln(ans[92],' ', ans[95],
                  ' ', ans[98])
end.
```

[1] См. выше структуру обрабатываемой строки с информацией об очередной марке бензина на той или иной АЗС.

“В поисках поиска”: задачи ЕГЭ, посвященные поиску информации на сайтах

О.Б. Богомолова, д. п. н., учитель информатики и математики ГОУ СОШ № 1360, Восточный округ г. Москвы

Д.Ю. Усенков, ст. н. с. Института информатизации образования Российской академии образования, Москва

Среди задач, предлагаемых на Едином государственном экзамене, можно выделить задачи, касающиеся определения объемов найденной информации (количества сайтов) при поиске в сети Интернет. Такие задачи затрагивают сразу две темы курса “Информатика и ИКТ” — “Количество информации” и “Сложные поисковые запросы” и нередко вызывают трудности у школьников.

При этом следует отметить, что с 2011 года такие задачи стали существенно сложнее. Если во всех предыдущих демовариантах ЕГЭ в подобных задачах требовалось только определить соотношение количеств ссылок, найденных по каждому из предложенных в задаче поисковых запросов (задача на ранжирование), то в 2011 году в демонстрационном варианте ЕГЭ, равно как и в тренажных контрольных по ЕГЭ, проводимых в течение 2010/11 учебного года, в подобной задаче уже ставится вопрос о вычислении количества найденных ссылок.

Для начала, как обычно, приведем весь перечень задач рассматриваемого типа, предлагавшихся в демонстрационных вариантах ЕГЭ с 2005 по 2011 год (для каждой задачи указан год и через тире — номер задачи).

2005 — В8. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

Для обозначения логической операции “ИЛИ” в запросе используется символ “|”, а для логической операции “И” — “&”.

А	чемпионы | (бег & плавание)
Б	чемпионы & плавание
В	чемпионы | бег | плавание
Г	чемпионы & Европа & бег & плавание

Ответ: ГБАВ.

2006 — В8. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

Для обозначения логической операции “ИЛИ” в запросе используется символ “|”, а для логической операции “И” — символ “&”.

А	разведение & содержание & меченосцы & сомики
Б	содержание & меченосцы
В	(содержание & меченосцы) | сомики
Г	содержание & меченосцы & сомики

Ответ: АГБВ.

2007 — B8. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

Для обозначения логической операции “ИЛИ” в запросе используется символ “|”, а для логической операции “И” — “&”.

А	волейбол | баскетбол | подача
Б	волейбол | баскетбол | подача | блок
В	волейбол | баскетбол
Г	волейбол & баскетбол & подача

Ответ: ГВАБ.

2008 — B8. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

Для обозначения логической операции “ИЛИ” в запросе используется символ “|”, а для логической операции “И” — “&”.

А	физкультура
Б	физкультура & подтягивания & отжимания
В	физкультура & подтягивания
Г	физкультура | фитнес

Ответ: БВАГ.

2009 — B10. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

Для обозначения логической операции “ИЛИ” в запросе используется символ “|”, а для логической операции “И” — “&”.

1	принтеры & сканеры & продажа
2	принтеры & продажа
3	принтеры | продажа
4	принтеры | сканеры | продажа

Ответ: 1234.

2010 — B10. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

Для обозначения логической операции “ИЛИ” в запросе используется символ “|”, а для логической операции “И” — “&”.

№	Запрос
1	канарейки | щеглы | содержание
2	канарейки & содержание
3	канарейки & щеглы & содержание
4	разведение & содержание & канарейки & щеглы

Ответ: 4321.

2011 — B9. В языке запросов поискового сервера для обозначения логической операции “ИЛИ” используется символ “|”, а для логической операции “И” — символ “&”.

В таблице приведены запросы и количество найденных по ним страниц некоторого сегмента сети Интернет.

Запрос	Найдено страниц (в тысячах)
Крейсер | Линкор	7000
Крейсер	4800
Линкор	4500

Какое количество страниц (в тысячах) будет найдено по запросу *Крейсер & Линкор*?

Считается, что все запросы выполнялись практически одновременно, так что набор страниц, содержащих все искомые слова, не изменялся за время выполнения запросов.

Ответ: 2300.

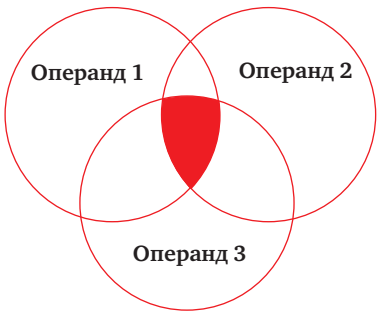
Начнем с более простых задач “старого типа”.

При их решении важно помнить простые правила:

- **операция “И” (&, ∧) *сокращает*** объем получаемого при поиске результата (уменьшает количество найденных сайтов), причем чем *больше* в ней задействовано *операндов*, тем *меньше* будет *объем* получаемого списка найденных сайтов;
- **операция “ИЛИ” (|, ∨) *увеличивает*** объем получаемого при поиске результата (увеличивает количество найденных сайтов), причем чем *больше* в ней задействовано *операндов*, тем *больше* будет *объем* получаемого списка найденных сайтов.

Запросы же, состоящие из одного-единственного операнда (ключевого слова) и не содержащие логических операций, можно рассматривать как запросы с операцией “ИЛИ” и с самым маленьким количеством операндов, т.е. в ранжированном списке такой запрос располагается непосредственно перед всеми остальными “ИЛИ”-запросами.

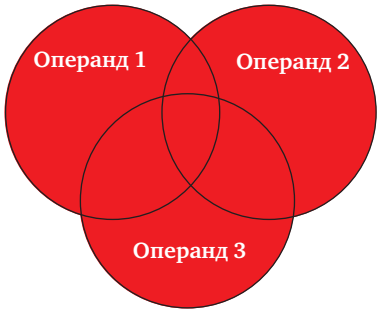
Причины этого очевидны: если операнды объединены при помощи операции “И”, то найденные документы (web-страницы) должны содержать *все* эти операнды (ключевые слова). Тогда, например, для запроса *операнд 1 & операнд 2 & операнд 3* получаем следующий результат:



Документ содержит:			Документ найден?
операнд 1	операнд 2	операнд 3	
			нет
+			нет
	+		нет
		+	нет
+	+		нет
	+	+	нет
+		+	нет
+	+	+	да

Если же операнды объединены при помощи операции “ИЛИ”, то найденные документы (web-страницы) могут содержать *хотя бы один* из этих операндов (ключевых слов). Тогда для запроса *операнд 1 | операнд 2 | операнд 3* результат будет таким:

Документ содержит:			Документ найден?
операнд 1	операнд 2	операнд 3	
			нет
+			да
	+		да
		+	да
+	+		да
	+	+	да
+		+	да
+	+	+	да



Таким образом, для приведенных в задаче поисковых запросов, включающих в себя только один вид логических операций (только “И” или только “ИЛИ”), можно сразу определить их место в формируемом списке ранжирования — если в задаче требуется расположить запросы по возрастанию[1] количества найденных документов, то:

- запросы с операцией “И” будут располагаться в начале списка, и чем больше в них задействовано операндов, тем такие запросы ближе к началу списка;
- запросы с операцией “ИЛИ” будут располагаться в конце списка, и чем больше в них задействовано операндов, тем такие запросы ближе к концу списка;
- запрос из одного-единственного ключевого слова будет располагаться в списке непосредственно перед запросами с операцией “ИЛИ”.

[1] Будьте внимательны! Разработчики вариантов ЕГЭ и тренажных работ вполне могут “подловить” учащихся, давая аналогичные задачи, в которых требуется, наоборот, выстроить поисковые запросы по порядку ***уменьшения*** количества найденных документов. Такие задачи решаются аналогично, но расположение запросов будет в списке обратным.

А что делать с поисковыми запросами, которые содержат оба типа логических операций — и “И”, и “ИЛИ”?

Таких задач в демонстрационных вариантах ЕГЭ было всего две, и в них можно было получить правильный ответ просто методом исключения: такой смешанный запрос располагается в ранжированном списке где-то посередине — между расположенным в его начале блоком запросов с “И” и расположенным в конце блоком запросов с “ИЛИ”.

Однако возможны и задачи, в которых смешанных запросов будет предложено несколько. В этом случае придется немного порассуждать, как ранжировать такие смешанные запросы между собой.

Возьмем для примера два смешанных запроса:

(кошки & собаки) | кролики

и

(кошки | собаки) & кролики

В первом случае будут найдены документы, в которых есть оба слова — “кошки” и “собаки”, и к ним будут добавлены *все* документы со словом “кролики”.

(кошки & собаки) | кролики

Во втором случае будут найдены документы, содержащие или слово “кошки”, или слово “собаки”, а из них будут отобраны только те документы, которые содержат также слово “кролики”.

(кошки | собаки) & кролики

То есть можно считать, что “действие” (уменьшение или увеличение количества найденных документов) логических операций “И” и “ИЛИ” “ослабляется”, когда они стоят в скобках, и “усиливается”, когда эти операции расположены вне скобок. То есть, когда операция “И” стоит в скобках, а “ИЛИ” — вне скобок, будет найдено больше документов, чем когда операция “ИЛИ” стоит в скобках, а “И” — вне скобок.

Вот теперь мы можем легко решить предложенные выше задачи из демовариантов ЕГЭ вплоть до 2010 года (для краткости будем далее повторять только таблицы с поисковыми запросами, так как формулировки условий задач одинаковы).

2005 — B8.

А	чемпионы | (бег & плавание)
Б	чемпионы & плавание
В	чемпионы | бег | плавание
Г	чемпионы & Европа & бег & плавание

Решение

Как мы уже говорили выше, запросы с операцией “И” в списке, ранжированном по возрастанию количества найденных документов, располагаются в самом начале, и чем больше в эти запросы включено операндов, тем они ближе к началу списка. Значит, наш список будут “открывать” запросы Г и Б.

Далее, — запросы с операцией “ИЛИ” располагаются в конце ранжированного списка, — значит, запрос В будет последним.

Ну а смешанный (по типу операций) запрос А, по методу исключения, нужно расположить на третьем месте в списке.

Ответ: ГБАВ.

2006 — B8.

А	разведение & содержание & мечеосцы & сомики
Б	содержание & мечеосцы
В	(содержание & мечеосцы) | сомики
Г	содержание & мечеосцы & сомики

Решение

Запросы А, Г и Б содержат операцию “И”, но разное количество операндов, — а мы помним, что “И”-запросы в ранжированном списке нужно располагать тем раньше, чем в них больше операндов. Поэтому начало списка — АГБ.

Оставшийся смешанный запрос В должен располагаться между “И”-запросами и “ИЛИ”-запросами. Но запросов с “ИЛИ” у нас нет, поэтому запрос В попросту завершает список.

Ответ: АГБВ.

2007 — B8.

А	волейбол | баскетбол | подача
Б	волейбол | баскетбол | подача | блок
В	волейбол | баскетбол
Г	волейбол & баскетбол & подача

Решение

Очевидно, что запрос Г, включающий в себя операции “И”, будет первым в формируемом списке.

Что же касается остальных запросов, включающих в себя операцию “ИЛИ”, то вспомним правило, что “ИЛИ”-запросы располагаются в конце списка, ранжированного по возрастанию количества найденных документов, и что чем больше в таких запросах используется операндов, тем ближе они к концу списка. Значит, первые три запроса надо выстроить в порядке ВАБ.

Ответ: ГВАБ.

2008 — B8.

А	физкультура
Б	физкультура & подтягивания & отжимания
В	физкультура & подтягивания
Г	физкультура | фитнес

Решение

Два запроса с операцией “И” будут располагаться в начале списка, причем первым будет идти запрос, содержащий больше всего операндов: БВ.

Запрос с операцией “ИЛИ” будет располагаться в конце списка.

Запрос же из одного-единственного слова “физкультура” будет, как мы уже говорили ранее, находиться непосредственно перед блоком запросов с “ИЛИ” (в нашем случае этот блок состоит из одного запроса).

Ответ: БВАГ.

2009 — B10.

1	принтеры & сканеры & продажа
2	принтеры & продажа
3	принтеры | продажа
4	принтеры | сканеры | продажа

Решение

Первыми в списке, очевидно, будут идти запросы с операцией “И”, причем (согласно количеству используемых в них операндов) они будут идти в порядке 1, 2.

Последними в списке будут стоять запросы с “ИЛИ”, и они (тоже согласно количеству операндов в них) будут следовать в порядке 3, 4.

Ответ: 1234.

2010 — B10.

1	канарейки | щеглы | содержание
2	канарейки & содержание
3	канарейки & щеглы & содержание
4	разведение & содержание & канарейки & щеглы

Решение

В начале списка надо поместить запросы с операцией “И”, причем чем больше операндов в них задействовано, тем ближе такой запрос расположен к началу списка. Значит, начало списка будет таким: 4, 3, 2.

Оставшийся же запрос с “ИЛИ”, очевидно, в списке будет самым последним.

Ответ: 4321.

А вот теперь перейдем к более сложной задаче с вычислением количества найденных по запросу документов.

2011 — B9. В языке запросов поискового сервера для обозначения логической операции “ИЛИ” ис-

пользуется символ “|”, а для логической операции “И” — символ “&”.

В таблице приведены запросы и количество найденных по ним страниц некоторого сегмента сети Интернет.

Запрос	Найдено страниц (в тысячах)
Крейсер | Линкор	7000
Крейсер	4800
Линкор	4500

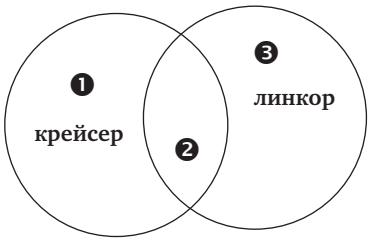
Какое количество страниц (в тысячах) будет найдено по запросу *Крейсер & Линкор*?

Считается, что все запросы выполнялись практически одновременно, так что набор страниц, содержащих все искомые слова, не изменялся за время выполнения запросов.

Решение

Подобные задачи можно, конечно, решать путем логических рассуждений. Но существует и универсальный способ их решения, который позволяет сделать решение всех таких задач типовым.

Построим для данной ситуации примерную диаграмму Венна (“примерную” — потому что мы в таких задачах *всегда* будем рисовать пересекающиеся области, хотя реально, как позже может выясниться по результатам вычислений, пересечение этих областей может и отсутствовать) и пронумеруем по порядку все получившиеся при этом “элементарные” области:



Теперь мы можем, считая порядковые номера выделенных областей своеобразными “переменными” (под значениями которых понимаются объемы соответствующих множеств, т.е. количества найденных документов), составить для каждого из указанных в таблице в условии задачи запросов систему уравнений. При этом не забываем, что, например, область “крейсер” состоит из двух выделенных нами “элементарных” областей — ❶ и ❷, точно так же, как область “линкор” состоит из двух “элементарных” областей ❷ и ❸.

$$\begin{cases} ❶ + ❷ + ❸ = 7000; \\ ❶ + ❷ = 4800; \\ ❷ + ❸ = 4500. \end{cases}$$

Аналогично, искомый запрос *Крейсер & Линкор* при этом трансформируется просто в значение “переменной” ❷.

Решать подобные системы уравнений учащиеся 10–11-х классов должны уметь. Например, можно подставить второе уравнение в первое и сразу получить значение “переменной” ❸:

$$❶ + ❷ + ❸ = 7000 \;\rightarrow\; 4800 + ❸ = 7000$$
$$❶ + ❷ = 4800$$
$$❸ = 7000 - 4800 = 2200.$$

Теперь, подставив полученное значение “переменной” ❸ в третье уравнение, можно получить значение интересующей нас “переменной” ❷:

$$❷ + ❸ = 4500 \;\rightarrow\; ❷ + 2200 = 4500$$
$$❸ = 2200$$
$$❷ = 4500 - 2200 = 2300.$$

Ответ: 2300.

Поскольку больше на данный момент подобных задач из демонстрационных вариантов ЕГЭ у нас нет (хотя наверняка они появятся в следующие годы!), для закрепления навыка решения таких задач воспользуемся заданиями из тренировочной работы № 2 по информатике, подготовленной МИОО и предлагавшейся в этом учебном году 11-классникам.

Вариант 1

B9. В языке запросов поискового сервера для обозначения логической операции “ИЛИ” используется символ “|”, а для логической операции “И” — символ “&”.

В таблице приведены запросы и количество найденных по ним страниц некоторого сегмента сети Интернет.

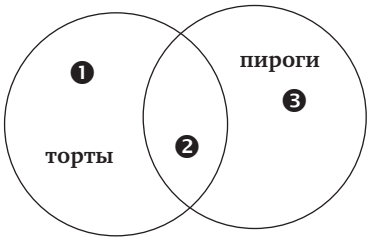
Какое количество страниц (в тысячах) будет найдено по запросу *Торты*?

Считается, что все запросы выполнялись практически одновременно, так что набор страниц, содержащих все искомые слова, не изменялся за время выполнения запросов.

Запрос	Найдено страниц (в тысячах)
Торты | Пироги	12 000
Торты & Пироги	6500
Пироги	7700

Решение

Вновь начинаем с рисования примерной диаграммы Венна (внешне она будет точно такой же, как и в предыдущей задаче):



Составляем систему уравнений, соответствующих указанным в таблице запросам:

$$\begin{cases} ❶ + ❷ + ❸ = 12\,000; \\ ❷ = 6500; \\ ❷ + ❸ = 7700. \end{cases}$$

Выражение же, соответствующее интересующему нас запросу *Торты*, выглядит так:

❶ + ❷.

Решаем систему уравнений. Для начала подставляем значение “переменной” ❷, известное нам по второму уравнению, в оба других уравнения системы, понижая тем самым ее размерность до двух:

$$\begin{cases} ❶ + 6500 + ❸ = 12\,000; \\ 6500 + ❸ = 7700. \end{cases}$$

Из второго уравнения получившейся системы нетрудно получить значение “переменной” ❸: оно равно 7700 – 6500 = 1200.

Подставив его в первое уравнение, получаем значение последней “переменной” — ❶:

❶ + 6500 + 1200 = 12 000 ⇒ ❶ = 4300.

Тогда нетрудно вычислить, что запросу *Торты* соответствует:

❶ + ❷ = 4300 + 6500 = 10 800 найденных документов.

Ответ: 10 800.

Вариант 2

B9. В языке запросов поискового сервера для обозначения логической операции “ИЛИ” используется символ “|”, а для логической операции “И” — символ “&”.

В таблице приведены запросы и количество найденных по ним страниц некоторого сегмента сети Интернет.

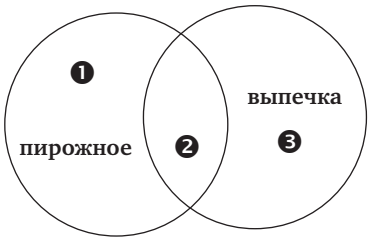
Какое количество страниц (в тысячах) будет найдено по запросу *Выпечка*?

Считается, что все запросы выполнялись практически одновременно, так что набор страниц, содержащих все искомые слова, не изменялся за время выполнения запросов.

Запрос	Найдено страниц (в тысячах)
Пирожное & Выпечка	5100
Пирожное	9700
Пирожное | Выпечка	14 200

Решение

Примерная диаграмма Венна:



Система уравнений, соответствующая заданным в таблице запросам:

$$\begin{cases} ❷ = 5100; \\ ❶ + ❷ = 9700; \\ ❶ + ❷ + ❸ = 14\,200. \end{cases}$$

Интересующий нас запрос соответствует выражению:

❷ + ❸.

Решаем систему уравнений. Для этого подставляем значение “переменной” ❷ из первого уравнения во второе, вычисляем тем самым значение “переменной” ❶:

❶ + 5100 = 9700 ⇒ ❶ = 4600.

Затем подставляем полученное значение ❶ в третье уравнение и вычисляем значение выражения для запроса *Выпечка*:

❷ + ❸ = 14 200 + 4600 = 9600.

Ответ: 9600.

А “на закуску” рассмотрим еще одну задачу того же типа, но несколько усложненную: в ней присутствует не два, а три ключевых слова[2].

Задача. Некоторый сегмент сети Интернет состоит из 1000 сайтов. Поисковый сервер в автоматическом режиме составил таблицу ключевых слов для сайтов этого сегмента. Вот ее фрагмент:

Ключевое слово	***Количество сайтов, для которых данное слово является ключевым***
сканер	*200*
принтер	*250*
монитор	*450*

Сколько сайтов будет найдено по запросу

(принтер | сканер) & монитор

если по запросу *принтер | сканер* было найдено 450 сайтов, по запросу *принтер & монитор* — 40, а по запросу *сканер & монитор* — 50 сайтов?

Решение

В этом случае вид диаграммы Венна будет другим — она будет состоять из трех кругов:

А далее, как и раньше, составляем систему уравнений для каждого заданного запроса (как в таблице, так и вне ее), не забывая также добавить условие, что всего сайтов было 1000:

$$\begin{cases} \textit{Сканер} & ❶ + ❷ + ❹ + ❺ = 200; \\ \textit{Принтер} & ❷ + ❸ + ❺ + ❻ = 250; \\ \textit{Монитор} & ❹ + ❺ + ❻ + ❼ = 450; \\ \textit{Принтер | сканер} & ❶ + ❷ + ❸ + ❹ + ❺ + ❻ = 450; \\ \textit{Принтер \&} & ❺ + ❻ = 40; \\ \textit{монитор} & ❹ + ❺ = 50; \\ \textit{Сканер \& монитор} & ❶ + ❷ + ❸ + ❹ + ❺ + ❻ + \\ \textit{Всего сайтов} & + ❼ = 1000. \end{cases}$$

[2] *http://sch556.narod.ru/doc/b9.doc.*

Интересующий нас запрос *(принтер | сканер) & монитор* при этом соответствует выражению:

❹ + ❺ + ❻.

Начинаем решать систему уравнений. Она, конечно, сложнее предыдущих, но при внимательном рассмотрении можно найти способ подстановки одних уравнений в другие, быстро приводящий к решению.

Вычтем из четвертого уравнения первые два:

❶ + ❷ + ❸ + ❹ + ❺ + ❻ –

– (❶ + ❷ + ❹ + ❺) – (❷ + ❸ + ❺ + ❻) =

= 450 – 200 – 250 = 0 ⇒ – ❷ – ❺ = 0 ⇒

⇒ ❷ + ❺ = 0.

Теперь вспомним, что значения наших "переменных", обозначенных цифрами в кружочках, это *количества найденных документов* (т.е. числа натурального ряда), *которые не могут быть отрицательными*. Следовательно, если сумма таких "переменных" равна нулю, то *все* переменные, входящие в эту сумму (в данном случае — ❷ и ❺), также "обязаны" равняться нулю. Следовательно, получаем:

❷ = 0; ❺ = 0.

Кстати, это и объяснение тому, что мы ранее называли нашу диаграмму Венна "примерной": равенство нулю этих двух "переменных" означает, что запросу *принтер & сканер* не соответствует ни один документ (сайт), то есть наша диаграмма Венна на самом деле должна выглядеть так:

А теперь сложим пятое и шестое уравнения, одновременно подставляя уже известное нам нулевое значение "переменной" ❺:

❺ + ❻ + ❹ + ❺ = 40 + 50 ⇒ ❹ + ❻ = 90.

Нетрудно догадаться, что (опять-таки с учетом нулевого значения "переменной" ❺) это и есть значение искомого нами выражения для запроса *(принтер | сканер) & монитор*, — следовательно, задача решена (хотя при необходимости, "расплетая" систему уравнений дальше, можно было бы вычислить значения всех используемых нами "переменных" — от ❶ до ❼ — и определить количества найденных документов для *любого* запроса с указанными ключевыми словами).

Ответ: 90.

Практические советы по поиску информации в Интернете

1. Различные поисковые сервисы ("Яндекс", Google, "Апорт", Yahoo и др.) формируют свои базы данных о web-сайтах независимо друг от друга. Поэтому, если вы не нашли нужную вам информацию в одном поисковом сервисе, обязательно попробуйте повторить поиск в другом.

2. При вводе в поле поиска нескольких ключевых слов через пробел поисковые сервисы воспринимают их как объединяемые логической операцией И. Поэтому чем больше задано ключевых слов, тем меньше будет найдено страниц и тем точнее поиск.

3. Для более точного поиска воспользуйтесь *языком запросов*. На разных поисковых сервисах его команды могут несколько отличаться, но в большинстве своем они едины для всех. Например (более полный список можно найти на самом поисковом сервисе в разделе "Помощь"):

- знак "&" — логическая связка И (поиск документов, где обязательно есть все связанные ею слова);
- знак "|" — логическая связка ИЛИ (поиск документов, где есть хотя бы одно из слов);
- знак "плюс" (+) перед словом требует, чтобы это слово обязательно присутствовало в найденных документах;
- знак "минус" (–) перед словом указывает, что это слово должно отсутствовать;
- заключив несколько слов в кавычки, можно искать точную цитату (слова обязательно должны в документе стоять именно в указанном порядке, падеже, склонении и т.д.); запись внутри такой цитаты символа "звездочка" (*) позволяет пропустить слово (т.е. вместо звездочки может стоять любое слово при сохранении точности цитирования для остальных слов);
- восклицательный знак (!) перед словом указывает, что нужно искать его именно в заданном регистре (строчном / заглавном / с прописной буквы) — так, как это слово записано в строке поиска.

4. Опция поиска картинок — удобный инструмент для поиска иллюстраций (на "Яндексе" он, например, включается выбором гиперссылки "Картинки" над полем поиска, справа). Воспользовавшись при этом гиперссылкой "Расширенный поиск", можно указать требуемый размер картинок (отсечь слишком мелкие, выбрав опцию "большие"), графический формат, вид картинки (например, опция "клипарт" ищет картинки на белом фоне) и т.д.

5. Если нужно найти информацию о каком-либо техническом устройстве (скажем, сотовом телефоне), а вы не помните его точную марку, попробуйте поискать его изображение. Найдя похожую картинку, можно щелкнуть на гиперссылке с URL, выведенной рядом с "образцом" картинки, и сразу перейти на соответствующий сайт.

6. Если вы ищете какое-либо место в городе (например, дом по его номеру и названию улицы), то на "Яндекс-карте" можно воспользоваться опцией просмотра панорам (хотя они есть не для всех городов и улиц): например, с их помощью вы сможете визуально просмотреть весь путь (и запомнить "приметные" объекты на маршруте), чтобы потом без ошибок добраться до нужного места. А опция измерения расстояний и карта метро помогут вам оценить время поездки или пешего хода.

ЭТО ПОЛЕЗНО ЗНАТЬ

Гаджеты и виджеты

А.В. Диков, г. Пенза

С появлением и развитием Интернета возникают связанные с ним новые термины. Одними из последних новых слов являются *гаджет* и *виджет*.

Гаджет (от англ. *gadget* — безделушка, ерунда) — какое-либо приспособление или устройство, техническая новинка. Гаджетом можно считать любой цифровой прибор, достаточно небольшой, чтобы надеть на руку или подключить к ПК, КПК или смартфону[1]. Например, стильные цифровые часы — это гаджет. Пожалуй, самым известным гаджетом сегодня является интернет-планшет iPad от компании Apple.

Виджет (от англ. *widget* — “штуковина”, “штучка”) — любое полезное приспособление. В информатике виджет — небольшое приложение (мини-приложение) для решения отдельных задач или быстрого получения информации из Интернета без помощи браузера, интегрированное в какой-либо документ (например, на веб-страницу) или на рабочий стол операционной системы. Внешне виджеты имеют вид небольшого окна (формы), предоставляющего дополнительную информацию, например, прогноз погоды или курс валют.

Добавим, что иногда такое приложение также называют *гаджет* (см., например, два следующих абзаца).

Типичными примерами виджетов как мини-приложений являются Google Gadgets (существуют варианты для боковой панели Google Desktop и для веб-страницы или блога iGoogle).

Еще одним примером гаджетов не как устройств являются мини-приложения для боковой панели операционной системы Windows 7.

В Интернете можно встретить большое число виджетов для встраивания в свой блог или веб-страницу. Кратко опишем некоторые из них.

[1] Статья в Википедии — свободной энциклопедии (*http://ru.wikipedia.org/wiki/Гаджет*).

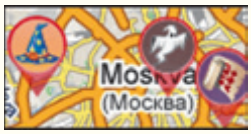
Поиск в Википедии
http://desktop.google.com/

Старинная китайская игра Маджонг
http://gadgets.project.googlepages.com/

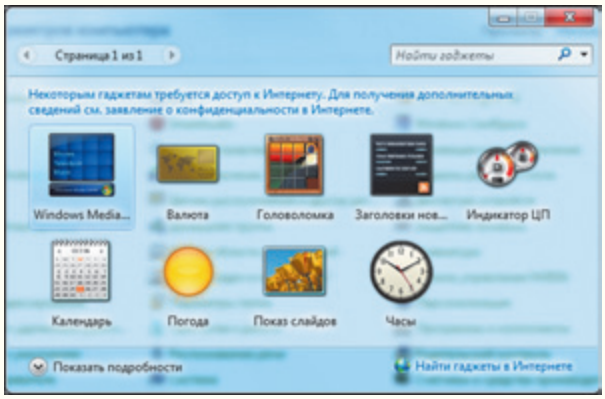
Курсы USD, EUR и кросс-курс EUR/USD
http://giss.ru/

Мировые часы
http://www.ljmsite.com/

Последние новости
http://www.google.com/



Москва: Тайны, легенды, аномальные зоны
http://sosedi-online.ru/



AudioPal (звук)

С помощью сервиса AudioPal [*http://www.audiopal.com/*] можно бесплатно, указав только адрес своей электронной почты, создать звуковой файл, который и будет выслан на указанный адрес.

Запускает начало процесса кнопка GET YOURS IT'S FREE!.

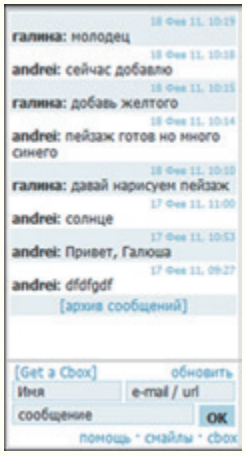
AudioPal поддерживает следующие способы создания звукозаписи:

- через телефонный звонок;
- через микрофон;
- набор текста, который система преобразует в звук (есть возможность выбора языка и голоса для прочтения текста);
- загрузка готового аудиофайла в формате mp3.

Всего за три шага можно подготовить и отправить на свой e-mail информацию о созданном звуковом файле. В высланном системой письме будет содержаться ссылка на сгенерированный виджет в следующем тексте “Click Here to pickup your AudioPal code” (“Щелкни здесь, чтобы взять код”). В открывшейся веб-странице можно увидеть изображение виджета и прослушать через него звукозапись. Красная кнопка “Copy” положит в буфер обмена код для встраивания виджета в блог или веб-страницу:

```
<img style="visibility:hidden;width:0px;
 height:0px;" border=0 width=0 height=0
 …
</object>
```

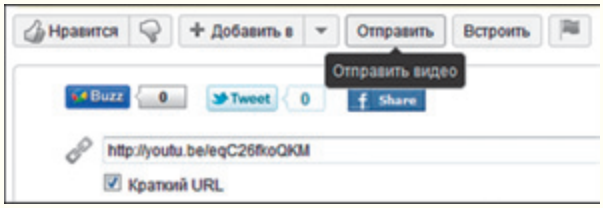
Cbox (чат)

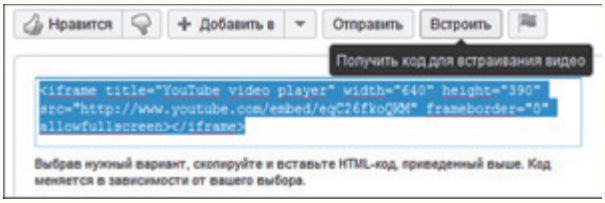


Cbox (от англ. *chat box* — оболочка для чата) [*http://cbox.ws/*] — чат для социальной сети, имеющий ряд возможностей: включает историю сообщений, хорошо настраивается под сайт, имеет элементы управления чатом. Бесплатная версия имеет меньше возможностей по сравнению с платной. Для получения кода виджета необходимо зарегистрироваться.

Социальные сети, предназначенные для обмена мультимедиа-материалами пользователей Интернета, предлагают код для встраивания размещенного мультимедиа в виде виджетов на веб-страницах или блогах клиентов или в других социальных службах.

YouTube (возможный перевод с англ. — “твоя труба/камера”) [*http://www.youtube.com/*] — сервис, предоставляющий возможность бесплатного размещения и просмотра любительского видео. Проект запущен в начале 2005 года Стивеном Ченом и Чедом Хёрли. Служба использует технологию Flash Video (flv), позволяющую получить хорошее качество записи при небольшом объеме передаваемых данных. Для размещения видеоматериала необходимо завести аккаунт. Сервис позволяет закачивать видео в различных форматах, но сразу преобразовывает его во флэш. YouTube позволяет через опцию “Встроить” взять html-код для внедрения видеоролика на свой веб-документ (блог, веб-страницу и т.п.). Опция “Отправить” дает только гиперссылку на видеоролик. Просмотр внедренного видео осуществляется в виде онлайн-трансляции.





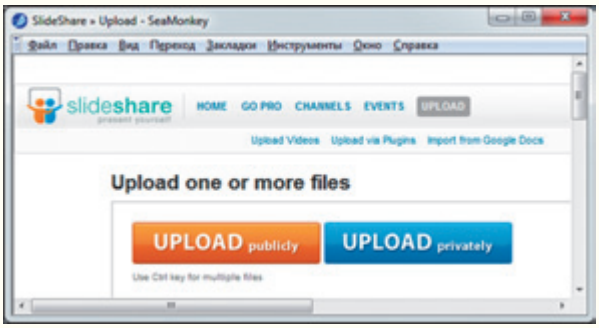
Сервис YouTube предоставляет даже некоторые инструменты редактирования размещенного видео: добавление звуковой дорожки, субтитров и аннотаций.

Issuu [*http://issuu.com/*] — англоязычный социальный сервис для создания и обмена публикациями (электронными книгами, брошюрами, журналами и т.д.). После загрузки на сервер документов происходит их конвертация в электронную книгу (для просмотра требуется плагин Flash-плеер). После конвертации, так же как и YouTube и SlideShare, сервис предоставляет html-код для внедрения в блог или любой другой веб-сайт. Issuu имеет достаточно удобный просмотр. Книгу в буквальном смысле можно “пролистать”. Существует удобное цифровое увеличение и перемещение страницы по экрану.

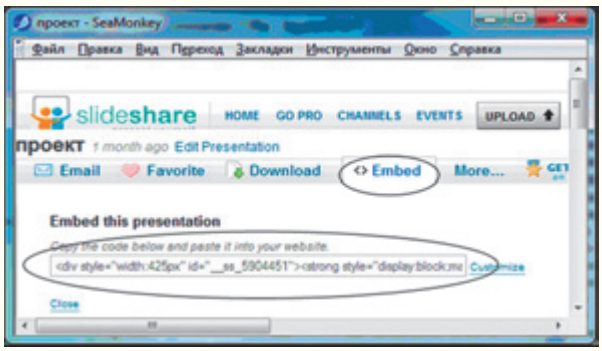
Конкурентным сервисом для обмена публикациями является англоязычный **Myebook** [*http://www.myebook.com/*].

Slideshare [*http://www.slideshare.net/*] (с англ. *slide share* — поделись слайдами) — служба, предназначенная для публикации презентаций, разработанных, например, в приложении PowerPoint, и просмотра их посетителями данного интернет-ресурса. Сервис Slideshare открылся в 2006 году. С его помощью появилась возможность быстрого обмена и распространения разработанных презентаций. Слайд-хостинг преобразует загруженные презентации в формат Flash, что позволяет просматривать видеоклипы в режиме реального времени (трансляция), уменьшить размер исходного файла и не запускать программу разработки презентаций.

Интерфейс ресурса пока не поддерживает русский язык, но это сильно не затрудняет работу с ним. Так, чтобы загрузить презентацию на сервер, следует нажать на кнопку “UPLOAD”:



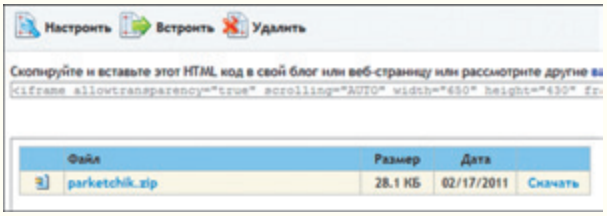
После загрузки и конвертации появится опция, по которой можно скопировать html-код и добавить его либо в свой блог, либо на свою веб-страницу или любой другой разрабатываемый веб-документ с тем, чтобы внедрить в него презентацию вместе с пультом управления (виджет).

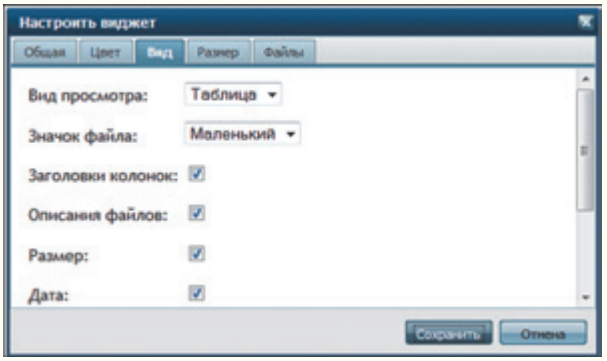


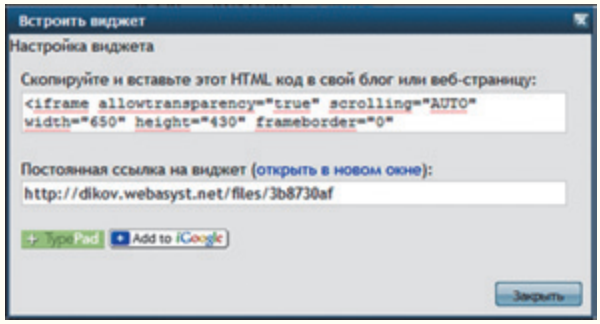
Вместо внедрения виджета презентации можно ограничиться добавлением гиперссылки на нее. Для этого на странице “My Uploads” (“Мои загрузки”) щелчком правой кнопкой мыши ссылка на презентацию копируется, а затем вставляется в свой веб-документ.

Похожими англоязычными сервисами для создания и обмена презентациями являются **Prezi** [*http://prezi.com/*] и **Slideboom** [*http://www.slideboom.com/*], документы **Google** с частично русским интерфейсом [*http://docs.google.com/*].

WebAsyst [*http://www.webasyst.ru/files/*] — русскоязычный сервис для хранения пользовательских файлов в Интернете, обмена файлами и совместной работы с файлами. Хранящиеся в архиве WebAsyst файлы можно отправить друзьям и коллегам, встроить с помощью виджетов на форумы, блоги и сайты. Настройка позволяет установить возможность добавления в папку файлов сторонних лиц. Сервис доступен после регистрации.







Использованные источники информации

1. Apple iPad. *http://devicegadget.ru/mediaplayers/apple-ipad/*

2. Porta News. *http://portanews.com/category/gadgets/*, 8.03.2011

3. *Диков А.* Веб 2.0. *http://www.sura.ru/dikov/resourses/webtwo.htm*

4. Интернет-магазин гаджетов. *http://www.berloga.su/collection/electronic-gadgets*

5. *Курвитс М.* Организация учебной работы при помощи блога. *http://blognauroke.blogspot.com/*

Ответы, решения, разъяснения к заданиям, опубликованным в газете “В мир информатики” ранее

1. Статья “Старинная и современная задача”

Напомним, что необходимо было, не составляя систему уравнений, решить следующую задачу: “46 человек собрались в туристский поход на лодках. Были приготовлены лодки двух видов: 6-местные и 4-местные. Сколько было лодок каждого вида, если все туристы разместились в 10 лодках и свободных мест не было?”.

Правильные ответы прислали:

— Борисов Валентин, средняя школа села Бабино, Удмуртская Республика, Завьяловский р-н, учитель **Мерзлякова Р.В.**;

— Гурьянова Анастасия, Кириченко Анастасия, Севастьянова Дарья и Ходюк Екатерина, Московская обл., г. Краснознаменск, Московский кадетский корпус “Пансион воспитанниц МО РФ”, учитель **Федорова Л.А.**;

— Джанхотова Тамара, г. Волгоград, поселок Горьковский, школа № 8, учитель **Брусенская М.С.**;

— Диков Андрей и Филимонова Галина, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Минакова Татьяна, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Новикова Светлана, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Тананаева Анастасия и Тананаева Ксения, Владимирская обл., г. Струнино, школа № 11, учитель **Волков Ю.П.**;

— Хомченко Станислав, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**

Решение

Если убрать все 4-местные лодки, заменив их 6-местными, то можно разместить 60 человек, то есть дополнительно 14 человек. А так как число мест в одной лодке увеличилось на 2, то эти 14 человек ранее размещались в $14/2 = 7$ лодках. Итак, 4-местных лодок было 7, а 6-местных — $10 - 7 = 3$. Проверим, получается ли общее число людей (46): $3 \times 6 + 7 \times 4 = 46$.

Можно также рассуждать так. Если убрать все 6-местные лодки, заменив их 4-местными, то можно разместить 40 человек, то есть на 6 человек меньше. А так как число мест в одной лодке уменьшилось на 2, то эти 6 человек ранее размещались в трех лодках (6/2). И в этом случае результат — тот же. Такой способ решения задачи предложили учащиеся Московского кадетского корпуса из г. Краснознаменска.

2. Статья “Шесть вопросов (вариант 3)”

Ответы

1. При создании Рыбинского водохранилища был полностью затоплен старинный город Молога.

2. Наиболее известные партии — Одетта–Одиллия, Аврора, Хозяйка Медной горы — исполнила балерина Майя Плисецкая.

3. Первая фонетическая система правописания украинского языка появилась благодаря альманаху “Русалка Днестровая”.

4. Символом Канады является клен (сахарный клен).

5. На флаге Китайской Народной Республики имеется 5 звезд (одна большая и четыре маленьких).

6. Игрушка, которая была изобретена в 1930 году, — бумажный самолетик. Ее автор — Джек Нортроп.

Ответы представили:

— Авраменко Наталья, г. Воронеж, школа № 5 им. К.П. Феоктистова, учитель **Чернышева И.А.**;

— Адамюк Анастасия, Грибанов Владлен, Дукач Светлана, Дюбарова Анастасия, Клименко Надежда, Кирсанова Алеся, Пирогов Егор, Романова Надежда, Сагитова Зильда, Смолярова Наталья, Шошина Екатерина, г. Лесосибирск Красноярского края, поселок Стрелка, школа № 8 им. Константина Филиппова, учитель **Лопатин М.А.**;

— Анциферов Дмитрий, Брюханова Анжела, Власова Ирина, Гаврилов Вячеслав, Галенко Кристина, Довгалёва Кристина, Ермолаева Анна, Захаров Роман, Золотова Ирина, Кирбижеков Иван, Ковалёва Дарья, Комлева Вероника, Корниенко Ирина, Меметова Вероника, Миронова Александра, Мурзаева Яна, Павлов Влад, Перфильев Вадим, Покаместова Дарья, Рубеко Максим, Саакян Карина, Старкова Анастасия, Сысоева Дарья, Хохлова Анастасия, Чередник Илья, Чернышенко Дмитрий, Шадрин Василий, Шапина Елена и Шестаков Дмитрий, Красноярский край, г. Канск, школа № 5, учитель **Павлова Н.Н.**;

— Базылев Юрий, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Богданова Л.М.**;

— Белакова Наталья, Пензенская обл., поселок Тамала, школа № 1, учитель **Пашина Н.Д.**;

— Буханов Василий, Григорьев Кирилл и Юхтенко Илья, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Гвоздева Анастасия, Зорихин Алексей, Перевалов Антон, Сухорученко Сергей и Шишкина Анастасия, Свердловская обл., г. Нижняя Салда, школа № 7, учитель **Зорихина Н.Ю.**;

— Григоренко Дмитрий, Есипова Мария, Ли Илларион и Яснова Дарья, средняя школа поселка Осиновка, Алтайский край, учитель **Евдокимова А.И.**;

— Джанхотова Тамара, г. Волгоград, поселок Горьковский, школа № 8, учитель **Брусенская М.С.**;

— Лазуренко Денис, Шапошников Александр и Сафин Сергей, Москва, кадетская школа-интернат № 5 “Преображенский кадетский корпус”, учитель **Сергеев С.А.**;

— Микулик Илья, г. Астрахань, школа № 33 им. Н.А. Мордовиной, учитель **Лепехина С.М.**;

— Неофитова Елена, средняя школа села Янтиково, Чувашская Республика, учитель **Неофитова Н.Н.**;

— Ноздрин Данил, Республика Башкортостан, г. Стерлитамак, школа № 17, учитель **Орлова Е.В.**;

— Остяков Петр, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Полюхович Максим и Ядзевичюс Стас, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Каликина Т.В.**;

— Селин Влад, средняя школа поселка Ерофей Павлович, Амурская обл., Сковородинский р-н, учитель **Краснёнкова Л.А.**;

— Тайбарей Нина, основная школа поселка Каратайка, Архангельская обл., учитель **Безумова В.А.**;

— Шадрина Юлия, Чувашская Республика, г. Канаш, Канашский педагогический колледж, преподаватель **Воеводина Р.В.**

Большинство приславших ответы представили развернутые комментарии к ним (в том числе с указанием использованных источников информации). Спасибо всем!

3. Задача “Пятеро друзей”

Ответ: Антон Мишин, Борис Хохлов, Вадим Тихонов, Григорий Чехов, Дмитрий Белкин.

Правильные ответы прислали:

— Адамюк Анастасия, Грибанов Владлен, Дукач Светлана, Дюбарова Анастасия, Клименко Надежда, Кирсанова Алеся, Пирогов Егор, Романова Надежда, Сагитова Зильда, Смолярова Наталья и Шошина Екатерина, г. Лесосибирск Красноярского края, поселок Стрелка, школа № 8 им. Константина Филиппова, учитель **Лопатин М.А.**;

— Базылев Юрий, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Богданова Л.М.**;

— Васильева Юлия, Республика Башкортостан, г. Стерлитамак, школа № 17, учитель **Орлова Е.В.**;

— Голик Екатерина, Тананаева Анастасия и Тананаева Ксения, г. Струнино Владимирской обл., школа № 11, учитель **Волков Ю.П.**;

— Григорьев Кирилл и Юхтенко Илья, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Зорихин Алексей, Свердловская обл., г. Нижняя Салда, школа № 7, учитель **Зорихина Н.Ю.**;

— Каюмов Адель и Каюмов Ансель, Республика Татарстан, г. Бавлы, гимназия № 4, учитель **Шафиков Н.Р.**;

— Ковалёва Дарья, Комлева Вероника, Павлов Влад и Шадрин Василий, Красноярский край, г. Канск, школа № 5, учитель **Павлова Н.Н.**;

— Лазуренко Денис, Шапошников Александр и Сафин Сергей, Москва, кадетская школа-интернат № 5 “Преображенский кадетский корпус”, учитель **Сергеев С.А.**;

— Мингалиев Руслан и Тухбатуллина Миляуша, Республика Татарстан, Актанышский р-н, село Актаныш, средняя школа № 2, учитель **Гилязова Г.М.**;

— Сорокина Анна, Совхозная средняя школа, Московская обл., Серебряно-Прудский р-н, поселок Успенский, учитель **Жарикова Е.Н.**;

— Тайбарей Нина, основная школа поселка Каратайка, Архангельская обл., учитель **Безумова В.А.**;

— Чекмезов Артем, г. Волгоград, поселок Горьковский, школа № 8, учитель **Брусенская М.С.**;

— Чернышев Антон, г. Воронеж, школа № 5 им. К.П. Феоктистова, учитель **Чернышева И.А.**

4. Головоломки с числом 2011

Ответы представили:

— Адамюк Анастасия, Грибанов Владлен, Дукач Светлана, Дюбарова Анастасия, Клименко Надежда, Кирсанова Алеся, Пирогов Егор, Романова Надежда, Сагитова Зильда, Смолярова Наталья и Шошина Екатерина, г. Лесосибирск Красноярского края, поселок Стрелка, школа № 8 им. Константина Филиппова, учитель **Лопатин М.А.**;

— Баур Марина, Мутовина Александра, Решитило Наталья и Санникова Карина, Красноярский край, г. Канск, школа № 5, учитель **Павлова Н.Н.**;

— Григорьев Кирилл и Юхтенко Илья, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Григоренко Дмитрий, Есипова Мария, Ли Илларион и Яснова Дарья, средняя школа поселка Осиновка, Алтайский край, учитель **Евдокимова А.И.**;

— Джанхотова Тамара, г. Волгоград, поселок Горьковский, школа № 8, учитель **Брусенская М.С.**;

— Евченко Мария, Кольтякова Анна, Семенова Наталья, Харитонова Елена и Шафиева Алина, Республика Башкортостан, г. Стерлитамак, школа № 17, учитель **Орлова Е.В.**;

— Лазуренко Денис, Шапошников Александр и Сафин Сергей, Москва, кадетская школа-интернат № 5 “Преображенский кадетский корпус”, учитель **Сергеев С.А.**;

— Неделяев Денис, средняя школа поселка Ерофей Павлович, Амурская обл., Сковородинский р-н, учитель **Краснёнкова Л.А.**;

— Печерский Александр, Москва, кадетская школа-интернат № 5 “Преображенский кадетский корпус”, учитель **Хлопков Г.К.**;

— Полюхович Максим, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Каликина Т.В.**

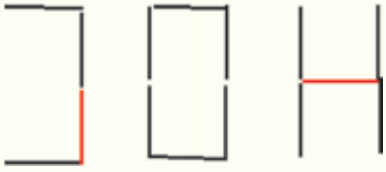
Правильные ответы

Задача 1. “Число-палиндром”

Переставив спички, оформленные красным цветом, и добавив спичку, выделенную синим, можно получить число-палиндром 202:

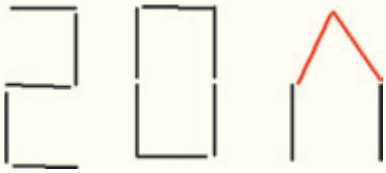
Задача 2. “Повесть Н.В. Гоголя”

Название повести “Нос” (прописными буквами) можно получить, переставив спички, оформленные красным цветом, и посмотрев на полученные буквы “с другой стороны”:



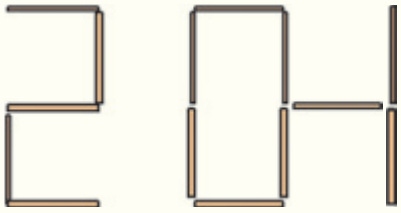
Задача 3. “Спортивная игра”

Из спичек можно получить слово “ГОЛ”:



Задача 4. “Любимое место отдыха”

Ответ показан на рисунке ниже (прочитайте его самостоятельно).



Александр Печерский из московской школы-интерната № 5 “Преображенский кадетский корпус” заметил, что можно получить также любимый *способ* отдыха — СОН ☺.

5. Числовой ребус “Лестница из слов”

Напомним, что необходимо было решить числовой ребус:

					Д
				А	**Д**
+			**Л**	**А**	**Д**
		К	**Л**	**А**	**Д**
	О	**К**	**Л**	**А**	**Д**
Д	**О**	**К**	**Л**	**А**	**Д**
9	8	7	6	5	4

Решение

Прежде всего видно, что **Д** = 4 или **Д** = 9. Первый вариант не подходит, так как нет таких значений цифры **А**, при которых сумма **5**А и числа 2 (которое в этом случае переходит “в уме” из последнего разряда) оканчивается на **А**.

Итак, **Д** = 9. Подставим известное значение в ребус:

					9
				А	9
			Л	**А**	9
		К	**Л**	**А**	9
	О	**К**	**Л**	**А**	9
9	**О**	**К**	**Л**	**А**	9
9	8	7	6	5	9

Из того, что из последнего разряда в разряд десятков “в уме” переходит 5, следует, что **А** — четная цифра. Значения 2 и 6 не подходят, так как при них суммы 4**Л** + 1 и 4**Л** + 3 нечетны и не могут оканчиваться на 6.

Проанализируем вариант **А** = 4. В этом случае 4**Л** + 2 оканчивается на 6 при **Л** = 1 или при **Л** = 2. При **Л** = 1 сумма трех цифр **К** не может быть равна 7. При **Л** = 6 указанная сумма равна 7, если **К** = 5. Но тогда из разряда тысяч в соседний слева разряд переходит 1, что не позволяет получить 8 в разряде десятков тысяч.

Итак, **А** = 8:

					9
				8	9
			Л	8	9
		К	**Л**	8	9
	О	**К**	**Л**	8	9
9	**О**	**К**	**Л**	8	9
9	8	7	6	5	9

Видно, что **Л** = 3 (цифра 8 уже использована):

					9
				8	9
			3	8	9
		К	3	8	9
	О	**К**	3	8	9
9	**О**	**К**	3	8	9
9	8	7	6	5	9

Итак, число **ДОКЛАД** = 987389.

Ответы прислали:

— Адамюк Анастасия, Грибанов Владлен, Дукач Светлана, Дюбарова Анастасия, Клименко Надежда, Кирсанова Алеся, Пирогов Егор, Романова Надежда, Сагитова Зильда, Смолярова Наталья и Шошина Екатерина, г. Лесосибирск Красноярского края, поселок Стрелка, школа № 8 им. Константина Филиппова, учитель **Лопатин М.А.**;

— Базылев Юрий, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Богданова Л.М.**;

— Баур Марина, Грибинец Саша, Кожевина Татьяна и Надёжин Максим, Красноярский край, г. Канск, школа № 5, учитель **Павлова Н.Н.**;

— Владимиров Виталий, Емелюкова Виктория, Петрова Алена, Семенова Екатерина и Яковлева Анастасия, основная школа села Именево, Республ-

лика Чувашия, Красноармейский р-н, учитель **Тимофеева И.А.**;

— Евченко Мария, Кольтякова Анна, Семенова Наталья, Харитонова Елена и Шафиева Алина, Республика Башкортостан, г. Стерлитамак, школа № 17, учитель **Орлова Е.В.**;

— Лазуренко Денис, Шапошников Александр и Сафин Сергей, Москва, кадетская школа-интернат № 5 "Преображенский кадетский корпус", учитель **Сергеев С.А.**;

— Минакова Татьяна, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Печерский Александр, Москва, кадетская школа-интернат № 5 "Преображенский кадетский корпус", учитель **Хлопков Г.К.**;

— Полюхович Максим, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Каликина Т.В.**;

— Пыров Егор, Тананаева Анастасия и Тананаева Ксения, г. Струнино Владимирской обл., школа № 11, учитель **Волков Ю.П.**;

— Хомченко Станислав, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**;

— Чекмезов Артем, г. Волгоград, поселок Горьковский, школа № 8, учитель **Брусенская М.С.**;

— Чернышев Антон, г. Воронеж, школа № 5 им. К.П. Феоктистова, учитель **Чернышева И.А.**

6. Задача "Потерянная гиря"

Напомним условие: "Было 8 гирь массами 1, 2, …, 8 кг без надписей. Все гири выполнены из одного материала, так что чем больше вес гири, тем больше ее размер (но при этом размер не пропорционален весу). Одну из гирь потеряли. Необходимо за два взвешивания на чашечных весах выяснить, какая именно гиря потеряна".

Ответы представили:

— Адамюк Анастасия, Грибанов Владлен, Дукач Светлана, Дюбарова Анастасия, Клименко Надежда, Кирсанова Алеся, Пирогов Егор, Романова Надежда, Сагитова Зильда, Смолярова Наталья и Шошина Екатерина, г. Лесосибирск Красноярского края, поселок Стрелка, школа № 8 им. Константина Филиппова, учитель **Лопатин М.А.**;

— Базылев Юрий, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Богданова Л.М.**;

— Диков Андрей и Филимонова Галина, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Минакова Татьяна, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Новикова Светлана, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Хомченко Станислав, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**

Решение

Пронумеруем имеющиеся гири в порядке возрастания размера. Произведем первое взвешивание: на левую чашку весов положим гири № 3 и № 4, а на правую — № 7. Прежде чем рассматривать возможные случаи состояния весов, заметим, что гиря № 3 может иметь вес 3 или 4 кг, гиря № 4 — 4 или 5 кг, гиря № 7 — 7 или 8 кг. Приведем также все возможные варианты сочетания этих гирь на разных чашках весов:

№ пп	Левая чашка весов		Правая чашка весов
	Гири весом	Общий вес	Гиря весом
1	3 и 4 кг	7 кг	7 кг
2	3 и 5 кг	8 кг	7 кг
3	4 и 5 кг	9 кг	7 кг
4	3 и 4 кг	7 кг	8 кг
5	3 и 5 кг	8 кг	8 кг
6	4 и 5 кг	9 кг	8 кг

Итак, возможные случаи.

1. Весы в равновесии. Это может быть при вариантах 1 и 5. Можно увидеть, что в первом случае веса гирь соответствуют их номерам, т.е. потеряна гиря весом 8 кг, а во втором — потеряна гиря весом 4 кг (так как гиря № 4 весит 5 кг). Чтобы определить действительно потерянную гирю, надо на одну чашку весов положить гирьки № 1 и № 3, а на вторую — № 4. Первая чашка перевесить не может. Если перевесит вторая чашка, то потеряли гирю весом 4 кг, в случае равновесия — весом 8 кг.

2. Если перевесит левая чашка весов, то, рассматривая варианты 3 и 6, можно установить, что потерянной может быть гиря весом 1, 2 или 3 кг. При втором взвешивании на первую чашку положим гири № 1 и № 2, а на вторую — № 3. Если перетянет первая чашка, то потеряна гиря весом 1 кг, если вторая — весом 3 кг, если весы в равновесии — весом 2 кг.

3. Если перевесит правая чашка весов, то потеряна одна из гирь весом 5, 6 или 7 кг. На одну чашку положим гири № 4 и № 7, на вторую — № 5 и № 6. Если перевесит первая чашка, то потеряли гирю весом 7 кг, если весы в равновесии — весом 6 кг, а если перевесит вторая чашка — весом 5 кг.

Юрий Базылев из школы № 1 поселка Надвоицы, Республика Карелия, описал вариант решения, в котором сначала сравниваются гири № 1 и № 6 (на одной чаше весов) и № 7 (на другой).

Редакция решила наградить всех перечисленных читателей дипломами. Поздравляем!

Ответы на задания, предложенные для самостоятельной работы в статье "Поиск наибольшего и наименьшего значений среди двух, трех и четырех чисел", прислали:

— Базылев Юрий, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Богданова Л.М.**;

— Минакова Татьяна, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Яновский Виталий, Москва, гимназия № 1530, учитель **Шамшев М.В.**

Издательский дом
первое сентября
новый этап развития

ЖУРНАЛ* «ИНФОРМАТИКА»

ПОДПИСКА НА ЭЛЕКТРОННУЮ ВЕРСИЮ ПРОДОЛЖАЕТСЯ!

ОЗНАКОМИТЕЛЬНЫЕ НОМЕРА И ОФОРМЛЕНИЕ ПОДПИСКИ –

НА САЙТЕ www.1september.ru

699 рублей – цена подписки для индивидуальных подписчиков и организаций за полгода
(в июле журнал не выходит)

ЭЛЕКТРОННАЯ ВЕРСИЯ

- Полностью соответствует бумажной
- Выходит гарантированно в срок
- Легко распечатывается на принтере
- Стоит существенно дешевле
- Доставляется по Интернету

* Внимание: со II полугодия 2011 года газета «Информатика» становится журналом.