Министерство образования республики Беларусь

Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

**Институт информационных технологий**

Специальность ИПОИТ

**КОНТРОЛЬНАЯ РАБОТА**

По курсу Основы защиты информации

Вариант № 15

Студент-заочник 2 курса

Группы № 680971

ФИО Никитина В.С.

Минск, 2017

Задание 1

Задание 1.1

База данных – это такая совокупность данных, которая организована в соответствии с определёнными правилами и имеющая определённую структуру, систематизированная таким образом, чтобы эти данные могли быть найдены и обработаны с помощью электронной вычислительной машины (ЭВМ). Она редактируется при помощи системы управления базами данных (СУБД).

СУБД – это совокупность языковых и программных средств, которая осуществляет доступ к данным, позволяет их создавать, менять и удалять.

Сервер баз данных - выполняет обслуживание и управление базой данных и отвечает за целостность и сохранность данных, а также обеспечивает операции ввода-вывода при доступе клиента к информации.

Например, рассмотрим базу данных крупного банка. В ней есть все необходимые сведения о клиентах, об их адресах, кредитной истории, состояние расчетных счетов, финансовых операциях и т.д. Доступ к этой базе данных имеется у достаточно большого количества сотрудников банка, но среди них вряд ли найдется такое лицо, которое имеет доступ ко всей базе полностью и при этом способно единолично вносить в нее произвольные изменения. Кроме данных, база содержит методы и средства, позволяющие каждому из сотрудников оперировать только с теми данными, которые входят в его компетенцию. В результате взаимодействия данных, содержащихся в базе, с методами, доступными конкретным сотрудникам, образуется информация, которую они потребляют и на основании которой в пределах собственной компетенции производят ввод и редактирование данных.

Основные функциональные характеристики:

* Организация данных;
* Обеспечение целостности данных;
* Импорт-экспорт данных;
* Связывание таблиц и обеспечение доступа к данным;
* Добавление и изменение данных;
* Обработка данных;
* Отбор данных с помощью запросов и предоставление данных пользователям БД в виде таблиц, графиков и отчетов;
* Интеграция с Интернет (публикация данных в Интернете и в корпоративной сети);
* Обеспечение безопасности данных (шифрование данных, защита паролем, ограничение доступа к БД или отдельным ее объектам);
* Средства печати.

Анализ угроз ИБ базы данных на сервере. В состав угроз ИБ базы данных на сервере входят:

А) Сбои и отказы в аппаратуре вычислительных средств (угроза доступности);

Б) Вирусы и иные деструктивные программные элементы, распространяемые с использованием систем телекоммуникаций, обеспечивающих связь с внешней средой или внутренние коммуникации распределенной системы баз данных (угроза целостности);

В) Нарушение конфиденциальности информации, т. е. использование информации, хранящейся в базе данных, лицами или процессами, которые не были определены владельцами информации (угроза конфиденциальности);

Г) Нарушение целостности информации, т. е. модификация или уничтожение информации для ее обесценивания путем утраты соответствия с состоянием моделируемых сущностей реального мира (угроза целостности);

Д) Непреднамеренные ошибки пользователей, системных администраторов и других лиц, обслуживающих базу данных и сервер (угроза доступности);

Е) Угрозы, вызванные воздействием на систему баз данных и ее компоненты объективных физических процессов или стихийно развивающихся природных явлений (угроза доступности).

Задание 1.2

Приоритеты угроз (приоритет 1 – самая важная, приоритет 5 – самая маловажная):

Угроза «А», приоритет 1, из-за частых сбоев и отказов в аппаратуре вычислительных средств материальный ущерб от них самый высокий;

Угроза «В», приоритет 2, при нарушении конфиденциальности информации, в зависимости от характера распространенных сведений, может быть предусмотрена дисциплинарная, административная и уголовная ответственность. Параллельно может быть рассмотрен вопрос и о привлечении к гражданско-правовой ответственности путем взыскания ущерба, в том числе и морального вреда. Также велик ущерб от потери репутации владельца в глазах пользователей;

Угроза «Г», приоритет 2, нарушение целостности информации, приводит к невозможности использовать информацию без восстановления. Помимо вероятности потерять важные данные, угрозе подвержена достоверность данных;

Угроза «Д», приоритет 3, непреднамеренные ошибки пользователей, системных администраторов и других лиц, вследствие их неопытности, встречаются часто, но реже, чем сбои и отказы в аппаратуре вычислительных средств, и устранения их последствий обходятся дешевле;

Угроза «Е», приоритет 3, объективные физические процессы или стихийно развивающиеся природные явления происходят очень редко, но влекут серьёзные последствия;

Угроза «Б», приоритет 4, проникновение ВПО в ЭВМ легко блокируется качественными антивирусными программами.

Задание 1.3

Способы защиты информации в системе базы данных (методы парирования угроз):

Угроза «А», метод «регламентация»: повышение надёжности ЭВМ у производителя, повышение качества ремонта и технического обслуживания ЭВМ;

Угроза «Б», метод «препятствие»: установка антивирусного программного обеспечения;

Угроза «В», метод «препятствие», метод «регламентация», метод «принуждение», метод «маскировка»: экранирование помещений, в которых находятся ЭВМ или дисплей, установка различных генераторов шума. Разработка специализированного программного обеспечения, цель которого - не позволить постороннему человеку, не авторизированному в системе, получать информацию из системы. Примером такого вида защиты может быть система различных паролей. Также использование криптографического способа защиты данных. Ужесточение комплекса административно-правовых и уголовно-правовых норм, устанавливающих ответственность за несанкционированное использование данных;

Угроза «Г», метод «препятствие»: обеспечение отказоустойчивости (резервирование, дублирование, зеркалирование оборудования и данных), обеспечение безопасного восстановления (резервное копирование и электронное архивирование информации);

Угроза «Д», метод «препятствие»: использование принципа минимально возможных привилегий для пользователей и администраторов предписывает, чтобы каждый пользователь базы данных оперировал с данными, используя наименьший из возможных набор привилегий, необходимых для выполнения конкретной функции;

Угроза «Е», метод «регламентация»: должны быть разработаны документы, регламентирующие действия участников процесса обработки данных (как пользователей, так и системных администраторов) при возникновении нештатных ситуаций. Персонал должен проходить регулярные инструктажи и тренинги по обучению действиям в нештатных ситуациях, с иерархией передачи данных.

Задание 2

Задание 2.2

Пусть для дополнительной звукоизоляции ЗСП, высота которого 3 м, а длина 12 м, используется кирпичная перегородка в 2,5 кирпича.

Тогда для стены при перегородке в полкирпича нам понадобится

12 000 мм : 250 мм = 48 кирпичей в длину,

3 000 мм : 65 мм = 46 кирпичей в высоту, а всего

48 \* 46 = 2 208 кирпичей на стену при перегородке в полкирпича.

Тогда для стены при перегородке в 1 кирпич нам понадобится

12 000 мм : 120 мм = 100 кирпичей в длину,

3 000 мм : 65 мм = 46 кирпичей в высоту, а всего

100 \* 46 = 4 600 кирпичей на стену при перегородке в 1 кирпич.

Следовательно, на стену при перегородке в 2 кирпича понадобится

4 600 \* 2 = 9 200 кирпичей.

Тогда для стены при перегородке в 2,5 кирпича понадобится

2 208 + 9 200 = 11 408 кирпичей.

Следовательно, стоимость дополнительной кирпичной кладки, усиливающей звукоизоляцию стены для обеспечения затухания Q информационного сигнала в стене на частоте 1000 Гц до уровня 67 дБ, составит (при курсе валюты на 30.09.17 1,9580 бел. рублей за 1 у.е.) и исходных данных из табл. 2

(0,6+0,03)\*11 408\*1,9580 = 14 072,2 бел. рублей.

Задание 2.3

ФИО студента - НИКИТИНА ВЛАДА СЕРГЕЕВНА

1. Шифруем ФИО с помощью шифра перестановок

НИК ИТИ НАВ ЛАД АСЕ РГЕ ЕВН А

КНИ ИИТ ВНА ДЛА ЕАС ЕРГ НЕВ А

КНИИИТВНАДЛАЕАСЕРГНЕВА

1. Шифруем ФИО с помощью шифрующей таблицы

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Н | К | Т | Н | В | А | А | Е | Г | Е | Н |  |  |  |
| И | И | И | А | Л | Д | С | Р | Е | В | А |  |  |  |

НКТНВААЕГЕНИИИАЛДСРЕВА

НК ТН ВА АЕ ГЕ НИ ИИ АЛ ДС РЕ ВА

1. Шифруем ФИО с помощью одиночной перестановки по ключу

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Б | Г | У | И | Р |  | Б | Г | И | Р | У |
| 1 | 2 | 4 | 3 | 5 | 1 | 2 | 3 | 4 | 5 |
| Н | И | А | Р | Н | Н | И | Р | Н | А |
| И | Н | Д | Г | А | И | Н | Г | А | Д |
| К | А | А | Е | \_ | К | А | Е | **\_** | А |
| И | В | С | Е | **\_** | И | В | Е | **\_** | С |
| Т | Л | Е | В | **\_** | Т | Л | В | **\_** | Е |
|  |  |  |  |  |  |  |  |  |  |

НИРНАИНГАДКАЕ\_АИВЕ\_СТЛВ\_Е

НИРНА ИНГАД КАЕ\_А ИВЕ\_С ТЛВ\_Е

1. Шифруем ФИО с помощью шифра Цезаря. Для этого выбираем ключ k как смещение вправо на k букв в алфавите, например, k=2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| А | Б | В | Г | Д | Е | Ё | Ж | З | И | Й | К | Л | М | Н | О | П | Р | С | Т | У | Ф | Х | Ц | Ч | Ш | Щ | Ъ | Ы | Ь | Э | Ю | Я |
| Н | И | К | И | Т | И | Н | А | В | Л | А | Д | А | С | Е | Р | Г | Е | Е | В | Н | А |  |  |  |  |  |  |  |  |  |  |  |
| П | К | М | К | Ф | К | П | В | Д | Н | В | Ё | В | У | Ж | Т | Е | Ж | Ж | Д | П | В |  |  |  |  |  |  |  |  |  |  |  |

ПКМКФ КПВДН ВЁВУЖ ТЕЖЖД ПВ

1. Шифруем ФИО с помощью шифровального листка Ришелье

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | Координаты прорезей по оси абсцисс | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Координаты прорезей по оси ординат | 1 | х |  | х | х |  |  |  | х | х |  |
| 2 |  |  | х | х | х | х | х | х | х |  |
| 3 | х | х |  |  |  |  | х |  | х | х |
| 4 | х | х |  |  |  | х | х | х | х |  |
| 5 |  |  |  | х | х |  |  |  |  |  |
| 6 |  |  | х | х |  |  |  |  |  |  |

Расположение прорезей (координаты крестиков, ключ шифрования): (1:1); (1:3); (1:4); (1:8); (1:9); (2:3); (2:4); (2:5); (2:6); (2:7); (2:8); (2:9); (3:1); (3:2); (3:7); (3:9); (3:10); (4:1); (4:2); (4:6); (4:7); (4:8); (4:9); (5:4); (5:5); (6:3); (6:4).

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Координаты прорезей по оси абсцисс | | | | | | | | | | |
| 1 | | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Координаты прорезей по оси ординат | 1 | Н |  | И | К |  |  |  | И | Т |  |
| 2 |  |  | И | Н | А | В | Л | А | Д |  |
| 3 | А | С |  |  |  |  | Е |  | Р | Г |
| 4 | Е | Е |  |  |  | В | Н | А | \_ |  |
| 5 |  |  |  | \_ | \_ |  |  |  |  |  |
| 6 |  |  | \_ | \_ | \_ |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Координаты прорезей по оси абсцисс | | | | | | | | | | |
| 1 | | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Координаты прорезей по оси ординат | 1 | **Н** | ь | **И** | **К** | е | н | - | **И** | **Т** | о |
| 2 | г | - | **И** | **Н** | **А** | **В** | **Л** | **А** | **Д** | - |
| 3 | **А** | **С** | - | ж | е | л | **Е** | - | **Р** | **Г** |
| 4 | **Е** | **Е** | ф | - | э | **В** | **Н** | **А** | \_ |  |
| 5 |  |  |  | \_ | \_ |  |  |  |  |  |
| 6 |  |  | \_ | \_ |  |  |  |  |  |  |

НЬИКЕН-ИТОГ-ИНАВЛАД-АС-ЖЕЛЕ-РГЕЕФ-ЭВНА

1. Шифруем ФИО с помощью маршрутов Гамильтона. Ключ и длина зашифрованных блоков соответственно равны: К=<2,1,1> (маршруты 2, 1, 1), L=4

Б1=<НИКИТИНА> Б2=<ВЛАДА\_СЕ> Б3=<РГЕЕВНА\*>

Для первого блока <НИКИТИНА> переставляем буквы по маршруту № 2, получаем ИАНКНИИТ. Для второго блока <ВЛАДА\_СЕ> переставляем буквы по маршруту № 1, получаем АДЛВА\_ЕС. Для третьего блока <РГЕЕВНА\*> переставляем буквы по маршруту № 1, получаем ЕЕГРВН\*А.

Объединяем блоки с переставленными буквами в сообщение:

Тl=<ИАНКНИИТАДЛВА\_ЕСЕЕГРВН\*А>

Разбиваем полученное сообщение на блоки длины L=4. Готовый шифротекст выглядит как:

Tj = <ИАНК НИИТ АДЛВ А\_ЕС ЕЕГР ВН\*А>

1. Шифруем фамилию с помощью алгоритма RSA, используя при этом числа p и q, а также открытый ключ d:

p = 3, q = 11, d = 7

Вычисляем n = 3\*11=33

1Н - 15

2И - 10

3К - 12

4И - 10

5Т - 20

6И - 10

7Н - 15

8А - 1

Y(l) = (157) (mod 33) = 170859375 (mod 33) = 27;  
Y(2) = (107) (mod 33) = 10000000 (mod 33) = 10;  
Y(3) = (127) (mod 33) = 35831808 (mod 33) = 12;

Y(4) = (107) (mod 33) = 10000000 (mod 33) = 10;

Y(5) = (207) (mod 33) = 1280000000 (mod 33) = 26;

Y(6) = (107) (mod 33) = 10000000 (mod 33) = 10;

Y(7) = (157) (mod 33) = 170859375 (mod 33) = 27;

Y(8) = (17) (mod 33) = 1 (mod 33) = 1.

Получено зашифрованное сообщение Y(i) = <27,10, 12, 10, 26, 10, 27, 1>.

Задание 2.4.6

8,296,306 Linear-time top-k sort method

Перевести на русский название и части Abstract и Claims найденного патента, описать по-русски сущность запатентованного ОПС.

РЕФЕРАТ (АННОТАЦИЯ)

Данное изобретение относится к алгоритму, который извлекает только k элементов данных, имеющих наибольшие (или наименьшие) значения ключа из набора данных (т. е. результаты top-k) за время, линейно пропорциональное размеру набора данных. Предлагаемый метод с использованием алгоритма находит результаты top-k с использованием минимальной (или максимальной) структуры массива k-размера, которая сохраняет элементы-кандидаты результатов top-k путем сканирования всех элементов в наборе данных только один раз. Другими словами, данное изобретение обеспечивает линейный метод top-k сортировки, который находит результаты top-k в течение времени, линейно пропорционального размеру набора данных (то есть по большей мере O (n)), тогда как обычные алгоритмы сортировки для поиска результатов top-k не могут найти результаты top-k за время, линейно пропорциональное размеру набора данных (т. е. по меньшей мере O (n log n)).

ПАТЕНТНАЯ ФОРМУЛА

1. Метод сортировки по методу линейного времени Top-k, который находит k элементов данных, имеющих наибольшие значения ключа из набора данных из n элементов за время, линейно пропорциональное размеру набора данных, чтобы обрабатывать запросы top-k, которые извлекают только результаты с наивысшим рангом, упорядоченные в порядке важности в крупных поисковых системах или распределенных системах, включают в себя следующие этапы:

считывание k элементов данных в последовательности из набора данных S из n элементов данных и удаление этих k элементов данных из S, чтобы они не были прочитаны снова (этап 1 );

инициализацию структуры мини-набора topKHeap, которая будет возвращена в результате предлагаемого метода , в виде пустого дерева и построит topKHeap, вставив k элементов данных, прочитанных на шаге 1, в topKHeap (шаг 2);

извлечение элемента данных e из набора данных S (этап 3);

сравнение значения ключа элемента e, извлеченного на шаге 3, с ключевым значением корневого узла r topKHeap (шаг 4);

замена значения ключа корневого узла r на элемент e, если ключевое значение e больше, чем значение r (этап 5 );

сравнение ключевого значения корневого узла r, которое заменено на шаге 5, с ключевыми значениями его дочерних узлов и корректировка topKHeap, так чтобы значение ключа родительского узла не превышало ключевые значения его дочерних узлов (этап 6);

удаление элемента e из набора данных S (этап 7);

повторение шагов с 3 по 7, пока набор данных S не станет пустым, так чтобы не было элемента, который нужно прочитать (этап 8);

возвращение topKHeap, который состоит из элементов данных, имеющих наибольшие значения ключа, в результате, когда набор данных S становится пустым (этап 9),

где на этапе 9 возвращают k элементов данных, хранящихся в topKHeap, в порядке убывания этих значений ключа 1) путем извлечения k элементов данных из корневого узла topKHeap и 2) путем их считывания в обратном порядке экстракции, при этом временная сложность метода представляется как сумма времени для инициализации topKHeap с использованием k элементов данных, первоначально выбранных (O (k log k)), времени для восстановления topKHeap с другими (n-k) элементами данных (O ( (n-k) log k)) и время для возврата k элементов данных в порядке убывания, в результате чего O (c 1 k log k + c 2 k log k + (n-k) log k) = O ( (n + ck) log k) (c = (c 1 + c 2 -1)) относительно констант c, c 1 и c 2 , где поскольку c и k являются константами, временная сложность становится O ((n + ck) log k) = O (n), так что временная сложность метода линейно пропорциональна числу всех элементов (т. е. n) набора данных.

2. Метод по п.1, где вся обработка, которая включает в себя считывание элементов данных в наборе данных S и удаление их из S, выполняется в линейном времени путем сканирования, когда набор элементов данных хранится в запоминающем устройстве, таком как диск.

3. Метод по п.1, где этап 2 создает начальное значение topKHeap в O (k log k) по времени 1) вставка нового элемента в качестве конечного узла topKHeap; 2) сравнение ключевых значений узлов с родительскими / дочерними отношениями по всем путям от вновь вставленного узла к корневому узлу; и 3) корректировка topKHeap так, чтобы значение ключа родительского узла не превышало ключевых значений его дочерних узлов.

4. Метод по п.1, на котором этап 4, переходит непосредственно на этап 7, чтобы удалить элемент e, если ключевое значение e меньше, чем значение корневого узла r.

5. Метод по п.1, в котором этап 6 выполняет шаг ReadjustHeap, так что topKHeap удовлетворяет ограничениям структуры минимального набора, так значение ключа родительского узла должно быть не больше, чем значение его дочерних узлов, когда ограничения не выполняются, в котором шаг ReadjustHeap повторяется для всех узлов, имеющих родительское/дочернее отношение от конечных узлов к корневому узлу.

6. Метод по п.1, где этап 8 повторяет этапы с 3 по 7 столько раз, сколько (n-k), где (n-k) представляет количество элементов данных, за исключением k элементов данных, прочитанных на этапе 1 .

7. Метод сортировки по методу линейного времени Top-k, который находит k элементов данных, имеющих наименьшие значения ключей из набора данных из n элементов за время, линейно пропорциональное размеру набора данных, чтобы обрабатывать запросы top-k, которые извлекают только результаты с наивысшим рангом, упорядоченные в порядке важности в крупных поисковых системах или распределенных системах, включают в себя следующие этапы:

считывание k элементов данных в последовательности из набора данных S из n элементов данных и удаление этих k элементов данных из S, чтобы они не были прочитаны снова (этап 1);

инициализацию максимальной структуру набора topKHeap, которая будет возвращена в результате предлагаемого метода , в виде пустого дерева и построит topKHeap, вставив k элементов данных, прочитанных на шаге 1, в topKHeap (шаг 2);

извлечение элемента данных e из набора данных S (этап 3);

сравнение значения ключа элемента e, извлеченного на шаге 3, с ключевым значением корневого узла r topKHeap (шаг 4);

замена значения ключа корневого узла r на элемент e, если ключевое значение e меньше, чем значение r (шаг 5);

сравнение значения ключа корневого узла r, которое заменено на шаге 5, с ключевыми значениями его дочерних узлов и корректировка topKHeap, так что значение ключа родительского узла не меньше ключевых значений его дочерних узлов (этап 6) ;

удаление элемента e из набора данных S (этап 7);

повторение шагов с 3 по 7, пока набор данных S не станет пустым, так чтобы не было элемента, который нужно прочитать (этап 8);

возвращение topKHeap, который состоит из элементов данных, имеющих наименьшие значения ключа, в результате, когда набор данных S становится пустым (этап 9), где на этапе 9 возвращают k элементов данных, хранящихся в topKHeap, в порядке возрастания этих значений ключа 1) путем извлечения k элементов данных из корневого узла topKHeap и 2) путем их считывания в обратном порядке экстракции, при этом временная сложность метода представляется как сумма времени для инициализации topKHeap с использованием k элементов данных, первоначально выбранных (O (k log k)), времени для восстановления topKHeap с другими (n-k) элементами данных (O ( (n-k) log k)) и время для возврата k элементов данных в порядке возрастания, в результате чего O (c 1 k log k + c 2 k log k + (n-k) log k) = O ( (n + ck) log k) (c = (c 1 + c 2 -1)) относительно констант c, c 1 и c 2 , где поскольку c и k являются константами, временная сложность становится O ((n + ck) log k) = O (n), так что временная сложность метода линейно пропорциональна числу всех элементов (т. е. n) набора данных.

8. Метод по п.7, где вся обработка, которая включает в себя считывание элементов данных в наборе данных S и удаление их из S, выполняется в линейном времени путем сканирования, когда набор элементов данных хранится в запоминающем устройстве, таком как диск.

9. Метод по п.7, где этап 2 создает начальное значение topKHeap в O (k log k) по времени 1) вставляет новый элемент в качестве конечного узла topKHeap; 2) сравнивает ключевые значения узлов с родительскими/дочерними отношениями по всем путям от вновь вставленного узла к корневому узлу; и 3) корректирует topKHeap так, чтобы ключевое значение родительского узла не было меньше ключевых значений его дочерних узлов.

10. Метод по п.7, где этап 4 , непосредственно переходит к этапу 7, чтобы удалить элемент e, если ключевое значение e больше, чем значение корневого узла r.

11. Метод по п.7, где этап 6 выполняет шаг ReadjustHeap, так что topKHeap удовлетворяет ограничениям структуры набора max, что значение ключа родительского узла должно быть не меньше, чем значение его дочерних узлов, когда ограничения не выполняются, в котором шаг ReadjustHeap повторяется для всех узлов, имеющих родительское / дочернее отношение от конечных узлов к корневому узлу.

12. Метод по п.7, в котором этап 8 повторяет этапы с 3 по 7 столько раз, сколько (nk), где (n-k) представляет количество элементов данных, за исключением k элементов данных, прочитанных на этапе 1 .

13. Непрерывный машиночитаемый носитель информации, содержащий исполняемую компьютером программу, которая при выполнении выполняет метод сортировки по первому типу линейного времени по п.1 формулы изобретения.

14. Непрерывный машиночитаемый носитель данных, содержащий исполняемую компьютером программу, которая при выполнении выполняет метод сортировки по первому типу с линейным временем по п.7.

Сущность запатентованного ОПС

Сущностью данного изобретения является создание эффективного алгоритма, который извлекает только k элементов данных, имеющих наибольшие (или наименьшие) значения ключа из набора данных (т. е. результаты top-k) за время, линейно пропорциональное размеру набора данных. Предлагаемый метод с использованием алгоритма находит результаты top-k с использованием минимальной (или максимальной) структуры массива k-размера, которая сохраняет элементы-кандидаты результатов top-k путем сканирования всех элементов в наборе данных только один раз.