БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Международный Институт Дистанционного Образования

Кафедра “Информационные системы и технологии”

Контрольная работа по дисциплине «Структуры и алгоритмы обработки данных»

Тема работы:

**«Алгоритмы на Си»**

Выполнил:   
студент 2 курса, гр. 41703120  
Реут Владислав Леонидович

Проверил: Кондратёнок Е.В.

Минск 2022

**ВВЕДЕНИЕ**

Язык программирования С (си) является одним из самых популярных и распространенных языков. Он представляет компилируемый язык программирования общего назначения со статической типизацией, разработанный в 1969—1973 годах в компании Bell Labs программистом Деннисом Ритчи (Dennis Ritchie).

Язык С нередко называют языком программирования "среднего уровня" или даже "низкого уровня", так как он сочетает элементы языков высокого уровня с функциональностью и производительностью ассемблера и работает близко к аппаратной части компьютера. В итоге мы можем манипулировать данными на низком уровне и при этом использовать высокоуровневые конструкции для управления работы программы.

Первоначально язык С предназначался для написания операционной системы Unix. Впоследствии Си стал одним из популярных языков, а его основной сферой применения стало системное программирование, в частности, создание операционных систем, драйверов, различных утилит, антивирусов и т.д. К слову сказать, Linux большей частью написан на Си. Однако только системным программированием применение данного языка не ограничивается. Данный язык можно использовать в программах любого уровня, где важны скорость работы и производительность. Так, мы можем писать с помощью Си и прикладные приложения, и даже веб-сайты (используя технологию CGI - Common Gateway Interface). Но, конечно, для создания графического интерфейса и веб-приложений, как правило, выбираются более подходящие инструменты и технологии, но тем не менее круг использования Си довольно широк. Это в немалой степени определило популярность языка. Например, в известном рейтинге языков программирования TIOBE язык С долгое время уверенно удерживает второе место.

Си является компилируемым языком, а это значит, что компилятор транслирует исходный код на Си в исполняемый файл, который содержит набор машинных инструкций. Но разные платформы имеют свои особенности, поэтому скомпилированные программы нельзя просто перенести с одной платформы на другую и там уже запустить. Однако на уровне исходного кода программы на Си обладают переносимостью, а наличие компиляторов, библиотек и инструментов разработки почти под все распространенные платформы позволяет компилировать один и тот же исходный код на Си в приложения под эти платформы.

Развитие Си оказало большое влияние в целом на развитие языков программирования. В частности, его синтаксис стал основой для таких языков как С++, С#, Java, PHP, JavaScript. Особо следует сказать про связь с C++. C++ напрямую произошёл от Си. Но впоследствии их развитие происходило отдельно друг от друга, и даже появилась несовместимость между ними. Стандарт C99 добавил в язык Си ряд конфликтующих с C++ особенностей. В итоге в настоящее время оба языка являются фактически самодостаточными и развиваются независимо.

**Алгоритм** ([лат.](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D1%82%D0%B8%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *algorithmi* — от имени среднеазиатского математика [Аль-Хорезми](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D1%8C-%D0%A5%D0%BE%D1%80%D0%B5%D0%B7%D0%BC%D0%B8)[[1]](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC#cite_note-1)) — конечная совокупность точно заданных правил решения некоторого класса задач или набор [инструкций](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), описывающих порядок действий исполнителя для решения определённой задачи.

**1. Быстрая сортировка**





**Сложность алгоритма:**

Для наилучшего случая **Ω = О(n log n);**

Для среднего случая **Θ = О(n log n)**;

Для худшего случая **О = О(n2)**;

**Вспомогательные функции для вех нижеприведенных сортировок:**

void fillIntRandom(int\* array, int size, int border)

{

srand(time(NULL));

for (int i = 0; i < size; ++i)

\*(array + i) = rand() % border;

}

void printIntArray(int\* array, int size, int offset)

{

char format[7];

sprintf(format, "%%%dd", offset);

for (int i = 0; i < size; ++i) {

printf(format, array[i]);

if (i != size - 1)

printf(",");

}

printf("\n");

}

void swapInt(int\* a, int\* b)

{

int t = \*a;

\*a = \*b;

\*b = t;

}

**Код быстрой сортировки:**

void qs(int\* arr, int first, int last)

{

int i = first;

int j = last;

int x = arr[(first + last) / 2];

do {

while (arr[i] < x)

{

i++;

}

while (arr[j] > x)

{

j--;

}

if (i <= j)

{

swapInt(&arr[i], &arr[j]);

i++;

j--;

}

} while (i <= j);

if (i < last)

{

qs(arr, i, last);

}

if (first < j)

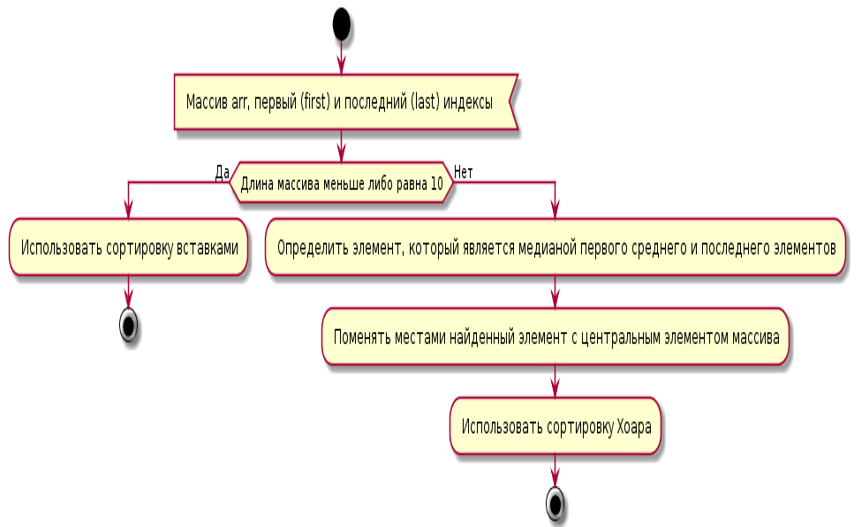
{

qs(arr, first, j);

}

}

**2. Улучшенная быстрая сортировка**

****

Для ее реализации нам необходима сортировка вставками.

Сортировку методом вставок рекомендуется использовать в случаях, когда необходимо отсортировать небольшое количество элементов или когда элементы исходной последовательности уже почти отсортированы

**Сложность алгоритма:**

Для наилучшего случая **Ω = О(n);**

Для среднего случая **Θ = О(n2)**;

Для худшего случая **О = О(n2)**;

**Блок-схема сортировки вставками:**

****

**Код сортировки вставками:**

void sortInserts(int\* arr, int len)

{

int temp, pos;

for (int i = 1; i < len; ++i)

{

temp = arr[i];

pos = i - 1;

while (pos >= 0 && arr[pos] > temp)

{

arr[pos + 1] = arr[pos];

pos--;

}

arr[pos + 1] = temp;

}

}

**Код улучшенной быстрой сортировки:**

int median(int\* a, int\* b, int\* c)

{

if (\*a > \*b)

{

int t = \*a;

\*a = \*b;

\*b = t;

}

if (\*b > \*c)

{

int t = \*b;

\*b = \*c;

\*c = t;

}

if (\*a > \*b)

{

int t = \*a;

\*a = \*b;

\*b = t;

}

return \*b;

}

void qsf(int\* arr, int first, int last)

{

if (last > 10)

{

int i = first;

int j = last;

int x = median(&arr[first], &arr[(first + last) / 2], &arr[last]);

do {

while (arr[i] < x)

{

i++;

}

while (arr[j] > x)

{

j--;

}

if (i <= j)

{

swapInt(&arr[i], &arr[j]);

i++;

j--;

}

} while (i <= j);

if (i < last)

{

qsf(arr, i, last);

}

if (first < j)

{

qsf(arr, first, j);

}

}

else

{

sortInserts(arr, last + 1); }

}

**3. Блочная сортировка**

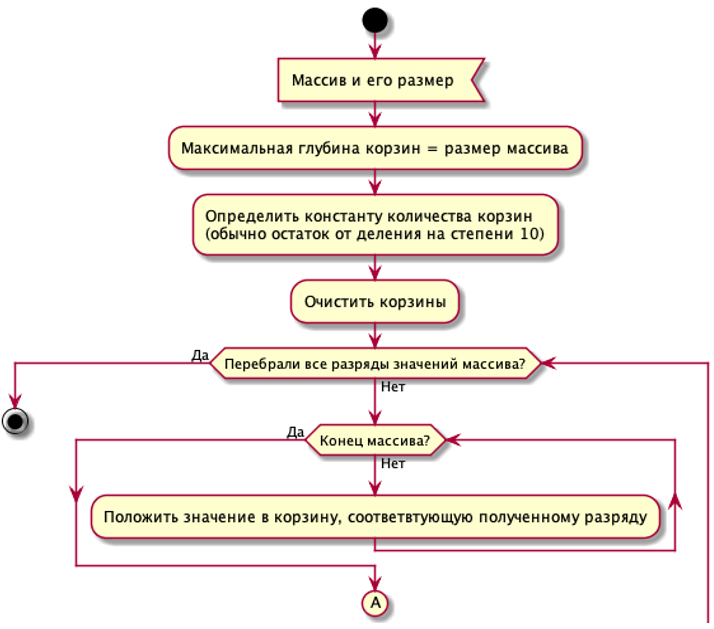
**Плюсы и минусы блочной сортировки.**

**Плюсы:**

- относится к классу быстрых алгоритмов с линейным временем исполнения O(N).

**Минусы:**

- сильно деградирует при большом количестве близких по значению элементов.

****

****

**Код блочной сортировки:**

void bucketSort2(int\* arr, int len)

{

const int max = len;

const int b = 10;

int buckets[b][max + 1];

for (int i = 0; i < b; ++i)

{

buckets[i][max] = 0;

}

for (int digit = 1; digit < 1000000000; digit \*= 10)

{

for (int i = 0; i < max; ++i)

{

int d = (arr[i] / digit) % b;

/\*

int counter = buckets[d][max];

buckets[d][counter] = arr[i];

counter++;

buckets[d][max] = counter;

\*/

buckets[d][buckets[d][max]++] = arr[i];

}

int idx = 0;

for (int i = 0; i < b; ++i)

{

for (int j = 0; j < buckets[i][max]; ++j)

{

arr[idx++] = buckets[i][j];

}

buckets[i][max] = 0;

}

}

}

**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Сайт Wikipedia.org
2. Сайт metanit.com