Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное учреждение высшего образования

«Пермский национальный исследовательский политехнический университет»

ПНИПУ

**Лабораторная работа**

**«Сортировки»**

Выполнил:

студент группы РИС-23-3б

Коротаев Александр Дмитриевич

Проверила:

доцент кафедры ИТАС

О.А. Полякова

2024 г.

**Разработка алгоритма**

**Постановка задачи:**

Реализовать поиски: линейный, бинарный, интерполяционный, линейный поиск строки, Бойера-Мура, Кнута-Морриса-Пратта.

**Анализ задачи:**

1. Создадим главный файл для основного кода**.**
2. Для удобства все поиски будут находиться в отдельных файлах.
3. Для использования поисков в главном файле будем подключать файлы с кодом поисков для дальнейшего использования.

Линейный поиск

int Line\_search(int\* mas, int& key)

{

int cnt = 0, index = -1;

for (int i = 0; i < \_msize(mas) / sizeof(mas[0]) && index == -1; i++)

{

if (mas[i] == key)

{

index = i;

}

}

return index;

# }

Бинарный поиск

int binary\_search(int\* mas, int& key)

{

int lenght\_mas = \_msize(mas) / sizeof(mas[0]);

int left = 0, right = lenght\_mas - 1, mid;

while (left <= right)

{

mid = (right + left) / 2;

if (mas[mid] == key)

{

return mid;

}

else

{

if (mas[mid] < key)

left = mid + 1;

else

right = mid - 1;

}

}

return -1;

# }

Интерполяционный поиск

int interpol\_search(int\* mas, int& key)

{

int lenght\_mas = \_msize(mas) / sizeof(mas[0]);

int left = 0, right = lenght\_mas - 1, mid, index = -1;

while (left <= right && index == -1)

{

mid = left + (((key - mas[left]) \* (right - left)) / (mas[right] - mas[left]));

if (mas[mid] == key)

{

index = mid;

}

else

{

if (mas[mid] < key)

left = mid + 1;

else

right = mid - 1;

}

}

return index;

# }

Линейный поиск подстроки в строке

int line\_str\_search(string& str, string& substr)

{

bool found\_str;

for (int i = 0; i <= str.length() - substr.length(); i++)

{

found\_str = true;

if (str[i] == substr[0])

{

for (int j = 1; j < substr.length() && found\_str; j++)

{

found\_str = (str[i + j] == substr[j]);

}

if (found\_str)

{

return i;

}

}

}

return -1;

# }

Поиск Бойера-Мура

void table\_formation(int\* shift\_table, string& substr)

{

for (int i = 0; i < 256; i++)

{

shift\_table[i] = substr.length();

}

for (int i = substr.length() - 2; i >= 0; i--)

{

if (shift\_table[int((unsigned char)substr[i])] == substr.length())

{

shift\_table[int((unsigned char)substr[i])] = substr.length() - i - 1;

}

}

}

int Boiere\_Moore(string& str, string& substr)

{

int shift\_table[256];

table\_formation(shift\_table, substr);

int index\_element = -1;

int current\_last\_char = substr.length() - 1;

while (current\_last\_char < str.length() && index\_element == -1)

{

bool flag = true;

for (int i = substr.length() - 1; i >= 0 && index\_element == -1 && flag; i--)

{

if (substr[i] != str[current\_last\_char - substr.length() + i + 1])

{

current\_last\_char += shift\_table[int((unsigned char)str[current\_last\_char - substr.length() + i + 1])];

flag = false;

}

else if (i == 0)

{

index\_element = current\_last\_char - substr.length() + 1;

}

}

}

return index\_element;

# }

Поиск Кнута-Морриса-Пратта

void prefix\_function(string& substr, int\* mas\_prefix)

{

int i = 1, j = 0;

while (i < substr.length())

{

if (substr[i] == substr[j])

{

mas\_prefix[i] = j++;

i++;

}

else

{

if (!j)

{

mas\_prefix[i] = 0;

i++;

}

else

{

j = mas\_prefix[j - 1];

}

}

}

}

int KMP(string& str, string& substr)

{

int\* mas\_prefix = new int[substr.length()];

mas\_prefix[0] = 0;

int str\_index = 0, substr\_index = 0, index\_element = -1;

prefix\_function(substr, mas\_prefix);

while (str\_index < str.length() && index\_element == -1)

{

if (substr[substr\_index] == str[str\_index])

{

str\_index++;

substr\_index++;

}

if (substr\_index == substr.length())

{

index\_element = str\_index - substr.length();

}

else if (str\_index < str.length() && substr[substr\_index] != str[str\_index])

{

if (!substr\_index)

{

str\_index++;

}

else

{

substr\_index = mas\_prefix[substr\_index - 1];

}

}

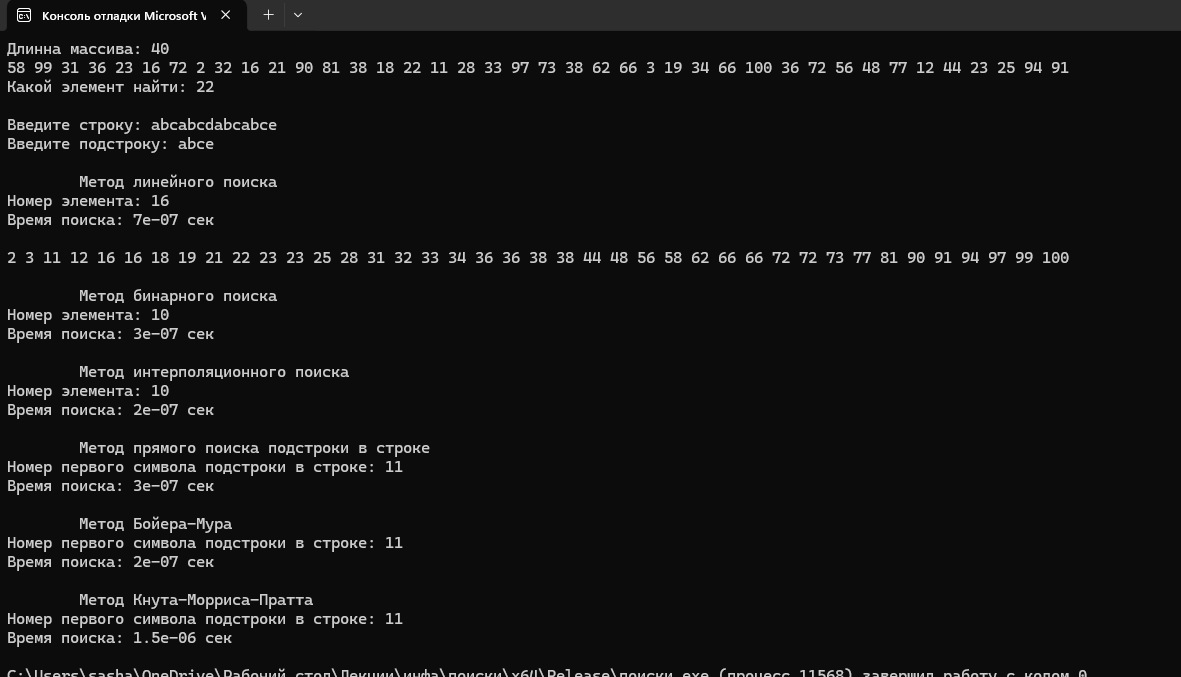
}

delete[] mas\_prefix;

return index\_element;

# }

**Пример решения**

****

**GitHub:** <https://github.com/Korovay4ik/Laboratory-works>