Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
**Высшая школа программной инженерии**

# КУРСОВАЯ РАБОТА

**Разработка структур данных**
по дисциплине: «Алгоритмы и структуры данных»

Выполнил
студент гр.B3530904/20030 В.В.Преловский

Руководитель
старший преподаватель, к.т.н. С.А. Федоров

«_____»_____ 202__ г.

Санкт-Петербург
2022

Содержание

Одна и таже инофрмация из входного файла вводится по разному в оперативную память с целью освоения работы с различными структурами данных - задание выполняется в виде 5 отдельных программных проектов , где необходимо использовать:

| | **Проект курсовой работыё** | | | | |
|---|---|---|---|---|---|
| **Средства** | 1 | 2 | 3 | 4 | 5 |
| массивы строк | - | + | - | - | - |
| массивы симоволов | - | + | - | - | - |
| внутренние процедуры головной программы | можно | + | + | - | - |
| Массив структур или структура массивов **выбрать** | - | - | + | + | - |
| файлы записей | - | - | + | + | - |
| модули | можно | можно | + | + | + |
| хвостовая рекурсия | - | - | - | + | + |
| однонаправленные списки заранее неизвестной длины | - | - | - | - | + |
| регуляное программирование | + | + | + | + | + |

Дан список сотрудников научно-исследовательской лаборатории в виде:

ФАМИЛИЯ ДОЛЖНОСТЬ

15 симв. 15 симв.

Пример входного файла:

Иванов техник

Отсортировать список в порядке повышения должности от "техника"до "вед. инженера". Пример выходного файла:

Иванов техник

Петров старший инженер

Цель работы - выбор структуры данных для решения поставленной задачи на современных микроархитектурах. Задачи:

1. Реализовать задание с использованием массивов строк.
2. Реализовать задание с использованием массивов символов.
3. Реализовать задание с использованием массива структур или структуры массивов.
4. Реализовать задание с использованием хвостовой рекурсии
5. Реализовать задание с использованием динамического списка
6. Провести аназил на регулярный доступ к памяти
7. Провести анализ на векторизацию кода
8. Провести сравнительный анализ реализаций

# 1 Реализация массива строк

Данной реализации было создано несколько массивов строк для фамилий, специальнойстей и для карьерного списка и две строки для сортировки.

```
redcharacter(SURNAME_LEN, redkind=CH_)                    ::
   tmpSurname = "", Surnames(AMOUNT) = ""
redcharacter(POST_LEN, redkind=CH_)                       ::
   tmpPost = "", Post(AMOUNT) = "",Doljnost(4) = ""
```

Сортировка производится методом пузырька. Поиск людей который необходимо поменять местами осуществляется следующим образом:

```
        redif (FindLoc(Doljnost,Post(j),reddim=1)>Findloc(
           Doljnost,Post(j+1),reddim =1)) redthen
            Swap = .redtrue.
            redelse redif (Surnames(j)>Surnames(j+1) .redand.
               Post(j) == Post(j+1)) redthen
               Swap = .redtrue.
        redend redif
```

Здесь же осуществляется регулярный доступ к данным.

# 2 Реализация массива символов

В данной реализации необходимо обратить на следующий фрагмент, согласно которому считанные фамиилии и долности будут храниться в "столбцах":

```
redcharacter(redkind=CH\_)                    ::  Surnames(
   SURNAME_LEN,AMOUNT) = ""
redcharacter(redkind=CH\_)                    ::  Post(Post_LEN,
   AMOUNT) = "",Doljnost(POST\_LEN,DOLJNOST\_LEN) = ""
```

Располагая данные таким образом мы предоставляем регулярный доступ к данным при сортривке. Сортировка производится так же методом пузырька, проверка производится таким образом:

```
        Swap = swap_doljnost(Doljnost,Post(:,j),Post(:,j+1))
        redif(.rednot. Swap .redand. redGT(Surnames(:,j),
           Surnames(:,j+1)) .redand. redAll(Post(:,j)==Post(:,j
           +1))) Swap = .redtrue.


    redpure redlogical redfunction swap_doljnost(Doljnost,arr1,
       arr2)
      redcharacter (redkind=CH_) arr1(:), Doljnost(:,:),arr2(:)
      redintent (redin) arr1,Doljnost,arr2
      redinteger i,j,k
      swap_doljnost = .redfalse.
      j=0
      k=0
      reddo  i=1, DOLJNOST\_LEN
```

```
        redif (redALL(arr1(:)==Doljnost(:,i)))j=i
        redif (redALL(arr2(:)==Doljnost(:,i)))k=i
      redend reddo
      swap_doljnost = j>k
  redend redfunction swap_doljnost

  redpure redlogical redfunction redGT(arr1, arr2)
   redcharacter(redkind=CH_), redintent(redin) :: arr1(:), arr2
      (:)

   redinteger :: i

   reddo i = 1, redMin(redSize(arr1), redSize(arr2)) - 1
      redif (arr1(i) /= arr2(i)) &
         redexit
   redend reddo
   redGT = arr1(i) > arr2(i)
redend redfunction redGT
```

Благодаря правильному расположению данных в массивах осуществляется регулярный доступ к памяти при перемещении фамилий

```
        redif (Swap) redthen
           tmpSurname         = Surnames(:,j+1)
           Surnames(:,j+1)    = Surnames(:,j)
           Surnames(:,j)      = tmpSurname

           tmpPost            = Post(:,j+1)
           Post(:,j+1)        = Post(:,j)
           Post(:,j)          = tmpPost
        redend redif
```

# 3 Реализация массива структур

# 4 Реализация рекурсивных процедуры в массиве структур

# 5 динамического списка

Глава Сравнение реализаций В данной главе будет проведен сравнительный анализ реализаций по критериям: регулярный доступ, векторизация,потенциальная векторизация.

Глава Вывод