ГУАП

КАФЕДРА № 34

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Жиданов К.А. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
|  |
| по курсу: |
| ЯЗЫКИ ПРОГРАММИРОВАНИЯ |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 3145 |  |  |  | Калинин Е.Д. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

**Вариант №9**

1. Найти медианную зарплату по выбранной категории

2. Найти значения зарплаты для заданного дециля

3. Построить гистограмму зарплаты для заданной категории (пол, профессия)

4. Отсортировать строки по заданной колонке

**Цель работы**

В .CSV файле хранятся значения зарплат. Требуется написать консольное приложение, которые считает данные и проведёт их обработку. Имя файла с данными и параметры задаются в командной строке.

**Ход работы**

Напишем программу:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

enum STATE { START, VALUE, END, ERROR };

typedef struct {

char name[9];

char surname[9];

char sex[9];

char postition[9];

unsigned int salary;

} nssps;

int main(int argc, char \*\*argv) {

FILE\* f = fopen("data.csv", "r");

char salary\_tmp[9];

nssps\* array = (nssps\*)malloc(sizeof(nssps)); //инициализация массива для считываемых данных

memset(&array[0], 0, sizeof(nssps));

memset(salary\_tmp, 0, sizeof(char));

enum STATE state = START; //инициализированная переменная состояний КА

char buffer[1024]; //буффер считывания

int string, i, j, param = 0, k = 0, n = 0;

int column = 0;

if (f == NULL) {

return 1;

}

while (!feof(f)) {

string = fread(buffer, 1, 1024, f);

for (i = 0; i < string; i++) {

if (state == ERROR)

break;

switch (state) { //конечный автомат для считанного буффера

case START:

if (buffer[i] == '\"') {

state = VALUE;

break;

}

else {

state = ERROR;

break;

}

case VALUE:

if (buffer[i] == '\"') {

state = END;

break;

}

switch (param) {

case 0:

array[n].name[k] = buffer[i];

break;

case 1:

array[n].surname[k] = buffer[i];

break;

case 2:

array[n].sex[k] = buffer[i];

break;

case 3:

array[n].postition[k] = buffer[i];

break;

case 4:

salary\_tmp[k] = buffer[i];

break;

}

if (param == 4) {

array[n].salary = atoi(salary\_tmp);

}

k++;

break;

case END:

if (buffer[i] == ',') {

state = START;

param++;

k = 0;

break;

}

else

if (buffer[i] == '\n') {

state = START;

n++;

array = (nssps\*)realloc(array, (n + 1) \* sizeof(nssps));

param = 0;

k = 0;

memset(&array[n], 0, sizeof(nssps));

memset(salary\_tmp, 0, 8\*sizeof(char));

break;

}

else {

state = ERROR;

break;

}

case ERROR:

break;

}

}

}

printf("enter column number to sort values by: ");

while ((column < 1) || (column > 5))

scanf("%d", &column);

for (i = 0; i < n - 1; i++){

switch (column) {

case 1:

for (j = 0; j < n - i - 1; j++)

if (strcmp(array[j].name, array[j + 1].name)>0) {

nssps temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

break;

case 2:

for (j = 0; j < n - i - 1; j++)

if (strcmp(array[j].surname, array[j + 1].surname) > 0) {

nssps temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

break;

case 3:

for (j = 0; j < n - i - 1; j++)

if (strcmp(array[j].sex, array[j + 1].sex) > 0) {

nssps temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

break;

case 4:

for (j = 0; j < n - i - 1; j++)

if (strcmp(array[j].postition, array[j + 1].postition) > 0) {

nssps temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

break;

case 5:

for (j = 0; j < n - i - 1; j++)

if (array[j].salary > array[j + 1].salary) {

nssps temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

break;

}

}

for (i = 0; i < n; i++) {

printf("%u: %s %s %s %s %u\n", i+1, array[i].name, array[i].surname, array[i].sex, array[i].postition, array[i].salary);

}

return 0;

}

При выполнении, программа выдает нужную зарплату для своего варианта.