#include <stdio.h>

#include <stdlib.h>

#include <dirent.h>

#include <string.h>

#include <math.h>

#define SECONDI\_MEDIA 2

int find\_range(char token[4][16][20]);

int main()

{

FILE \*dentro, \*fuori;

DIR \*d;

struct dirent \*dir;

int i, j, k, l, n\_files, flag, flag2, count, p\_avg, t\_avg, baseline, range;

char list[100][100], buff[1000], token[SECONDI\_MEDIA+1][16][20];

for (i=0;i<100;i++)

list[i][0] = '\0';

for (i=0;i<4;i++)

for (j=0;j<16;j++)

token[i][j][0] = '\0';

d = opendir("merged\_csv");

if (d)

{

for (i=0;(dir=readdir(d))!=NULL;) {

if(dir->d\_name[0] != '.') {

strcpy(list[i], dir->d\_name);

i++;

}

}

closedir(d);

n\_files = i;

printf("There are %d files in 'merged\_csv' folder\n", n\_files);

} else {

printf("ERROR! Directory not found.\n");

return 1;

}

mkdir("filtered");

for (i=0,count=0,flag2=0;i<n\_files;i++,count=0,flag2=0) {

sprintf(buff, "merged\_csv/%s", list[i]);

dentro = fopen(buff, "r");

sprintf(buff, "filtered/%s", list[i]);

fuori = fopen(buff, "w");

fgets(buff, 1000, dentro);

fprintf(fuori,"DATETIME;TIMESTAMP;DOORS;FIX;LONGITUDE;LATITUDE;SPEED;LINE;SHIFT;DEST;CURRENT;VEHICLE;P\_REF;TEMP;DELTA;RANGE\n");

for (flag=0;fgets(buff, 1000, dentro) != NULL && strlen(buff)>10;) {

for (j=SECONDI\_MEDIA;j>0;j--)

for (k=0;k<14;k++)

strcpy(token[j][k], token[j-1][k]);

for (j=0,l=0;j<14;j++,l++) {

for (k=0;buff[l]!=';'&&buff[l]!='\n';k++,l++)

token[0][j][k] = buff[l];

token[0][j][k] = '\0';

}

if (!atoi(token[0][2]))

flag = 1;

else if (flag) {

p\_avg = 0;

t\_avg = 0;

for (j=1;j<SECONDI\_MEDIA+1;j++) {

p\_avg += atoi(token[j][12]);

t\_avg += atoi(token[j][13]);

}

p\_avg = round(((float)p\_avg)/SECONDI\_MEDIA);

t\_avg = round(((float)t\_avg)/SECONDI\_MEDIA);

if (count == 0)

baseline = p\_avg;

range = find\_range(token);

sprintf(token[1][12], "%d", p\_avg);

sprintf(token[1][13], "%d", t\_avg);

sprintf(token[1][14], "%d", (p\_avg-baseline));

sprintf(token[1][15], "%d", range);

strcpy(buff, token[1][0]);

for (j=1;j<16;j++) {

strcat(buff, ";");

strcat(buff, token[1][j]);

}

fprintf(fuori, "%s\n", buff);

if (token[1][7][0] != '\0')

flag2 = 1;

count++;

flag = 0;

}

}

fclose(dentro);

fclose(fuori);

if (count < 10 || flag2 == 0) {

sprintf(buff, "filtered/%s", list[i]);

remove(buff);

} else

printf("File %2d of %d: %3d readings\n", i, n\_files, count);

}

system("pause");

return 0;

}

int find\_range(char token[4][16][20]) {

int i, n, min=1000000, max=0;

for (i=1;i<4;i++) {

n = atoi(token[i][12]);

if (n > max)

max = n;

if (n < min)

min = n;

}

return (max - min);

}