Bronson Pearl

Simulates weather data from taken in from a sensor. Outputs the data to screen, an output file, and reads it aloud using txt to speech.

**Mainwx.c**

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include <time.h>

#include "wx.h"

int main (void)

{

double wdata[WDATASZ];

//InitializeWeatherStation();

int dunits = InitializeWeatherStation(wdata);

time\_t now;

int logged;

while (1)

{

now = time(0);

UpdateWeatherStation(wdata);

logged = LogWeatherData("weatherdata.txt", now, dunits, wdata);

DisplayWeatherData(dunits, now, wdata);

SpeakWeatherData(dunits, now, wdata);

msDelay(WXDELAY);

}

return 0;

}

/\* UpdateWeatherStation(); taken out 13.3.2020

printf("\nEnter m for metric units, b for British units, or q to quit");

test = getch();

//InitializeWeatherStation();

if (test == 'b')

{

dunits = BRITISH;

}

if(test == 'q')

{

return 1;

}

DisplayWeatherData(dunits);\*/

**WX.c**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#include "wx.h"

void UpdateWeatherStation(double wreadings[WDATASZ])

{

printf("\nUpdating Weather Station\n");

wreadings[TEMPERATURE] = GetTemperature();

wreadings[HUMIDITY] = GetHumidity();

wreadings[PRESSURE] = GetPressure();

wreadings[WINDSPEED] = GetWindspeed();

wreadings[WINDDIRECTION] = GetWindDirection();

}

int InitializeWeatherStation(double wd[WDATASZ])

{

int iunits, i;

char test;

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

{

wd[i]= 0.0;

}

printf("Bronson's Tech104 Weather Station\n\n");

srand((unsigned)time(NULL));

// ask which unit they want

printf("\nEnter m for metric units, b for British units, or q to quit");

test = getch();

if (test == 'b')

{

iunits = BRITISH;

}

if (test == 'm')

{

iunits = METRIC;

}

else if(test == 'q')

{

exit(0);

}

return iunits;

}

/\* temperature = 25.1;

humidity = 55.4;

pressure = 101.7;

windspeed = 15.5;

direction = 320.0;

taken out 24.2.2020\*/

double GetTemperature(void)

{

return GetRandom(UPPERT-LOWERT)+LOWERT;

}

double GetHumidity(void)

{

return GetRandom(UPPERH-LOWERH)+LOWERH;

}

double GetPressure(void)

{

return GetRandom(UPPERP-LOWERP)+LOWERP;

}

double GetWindspeed(void)

{

return GetRandom(UPPERWS-LOWERWS)+LOWERWS;

}

double GetWindDirection(void)

{

return GetRandom(UPPERWD-LOWERWD)+LOWERWD;

}

void DisplayWeatherData(int units,time\_t wtime, double wreads[WDATASZ])

{

double /\*temperature,humidity,pressure,windspeed,direction,\*/ bt, bp, bw;

double creads[WDATASZ];

int i;

printf("\n%s",ctime(&wtime));

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

{

creads[i] = wreads[i];

}

if(units == METRIC)

{

printf("\nTemperature: %4.1lf C\tHumidity: %3.0lf%%\tPressure: %4.1lf kPa\n",creads[TEMPERATURE],creads[HUMIDITY],creads[PRESSURE]);

printf("Windspeed: %3.0lf km/hr\tDirection: %3.0lf Degrees\n\n",creads[WINDSPEED],creads[WINDDIRECTION]);

}

else

{

bt = (creads[TEMPERATURE] \* 9.0 / 5.0) + 32.0;

bp = creads[PRESSURE] \* KPATOINHG;

bw = creads[WINDSPEED] \* KMTOMPH;

printf("\nTemperature: %4.1lf F\tHumidity: %3.0lf%%\tPressure: %4.1lf inHg\n",bt,creads[HUMIDITY],bp);

printf("Windspeed: %3.0lf mph\tDirection: %3.0lf Degrees\n\n",bw,creads[WINDDIRECTION]);

}

}

int GetRandom(int range)

{

return rand() % range;

}

void msDelay(int ms)

{

long pause;

clock\_t now,then;

pause = ms\*(CLOCKS\_PER\_SEC/1000);

now = then = clock();

while( (now-then) < pause )

{

now = clock();

}

}

int LogWeatherData(char \* fname, time\_t wtime, int units, double wreads[WDATASZ])

{

FILE \*fp;

char ltime[TDATASZ];

fp = fopen(fname,"a");

if(fp==NULL)

{

printf("\ncannot open file. Data not retrieved.");

return 0;

}

strcpy(ltime, ctime(&wtime));

ltime[3] = ',';

ltime[7] = ',';

ltime[10]= ',';

ltime[19]= ',';

ltime[24] = '\0';

fprintf(fp,"%.24s,%d,%lf,%lf,%lf,%lf,%lf\n",

ltime,

units,

wreads[TEMPERATURE],

wreads[HUMIDITY],

wreads[PRESSURE],

wreads[WINDSPEED],

wreads[WINDDIRECTION] );

fclose(fp);

return 1;

}

int SpeakWeatherData(int units, time\_t wtime, double wreads[WDATASZ])

{

FILE \*fp;

char ltime[TSTRINGSZ];

struct tm \*info;

fp = fopen("wspeak.txt","a");

if(fp==NULL)

{

printf("\ncannot open file. Data not retrieved.");

return 0;

}

info = localtime(&wtime);

strftime(ltime,TDATASZ,"%A %B %Y %I %M %p", info);

if(units == METRIC)

{

fprintf(fp,"%s temperature %3.1lf degrees celsius humidity %3.0lf per cent pressure %4.1lf kilo pascals windspeed %3.0lf kilometers per hour wind direction %3.0lf degrees\n",

ltime,

wreads[TEMPERATURE],

wreads[HUMIDITY],

wreads[PRESSURE],

wreads[WINDSPEED],

wreads[WINDDIRECTION] );

}

else

{

fprintf(fp,"%s temperature %3.1lf degrees fahrenheit humidity %3.0lf per cent pressure %4.1lf inches of mercury windspeed %3.0lf miles per hour wind direction %3.0lf degrees\n",

ltime,

wreads[TEMPERATURE],

wreads[HUMIDITY],

wreads[PRESSURE],

wreads[WINDSPEED],

wreads[WINDDIRECTION] );

}

fclose(fp);

system("espeak -f wspeak.txt");

return 1;

}

**WX.h**

#ifndef WX\_H\_INCLUDED

#define WX\_H\_INCLUDED

// Constants

#define KMTOMPH 0.62137119

#define KPATOINHG 0.2953

#define METRIC 0

#define BRITISH 1

#define UPPERT 50

#define LOWERT -50

#define UPPERH 100

#define LOWERH 0

#define UPPERP 115

#define LOWERP 85

#define UPPERWS 130

#define LOWERWS 0

#define UPPERWD 360

#define LOWERWD 0

#define WXDELAY 5000

#define TEMPERATURE 0

#define HUMIDITY 1

#define PRESSURE 2

#define WINDSPEED 3

#define WINDDIRECTION 4

#define WDATASZ 5

#define TDATASZ 26

#define TSTRINGSZ 80

//function prototype

void UpdateWeatherStation(double wreadings[WDATASZ]);

int InitializeWeatherStation(double wd[WDATASZ]);

void DisplayWeatherData(int units,time\_t wtime, double wreads[WDATASZ]);

double GetTemperature(void);

double GetHumidity(void);

double GetPressure(void);

double GetWindspeed(void);

double GetWindDirection(void);

int GetRandom(int range);

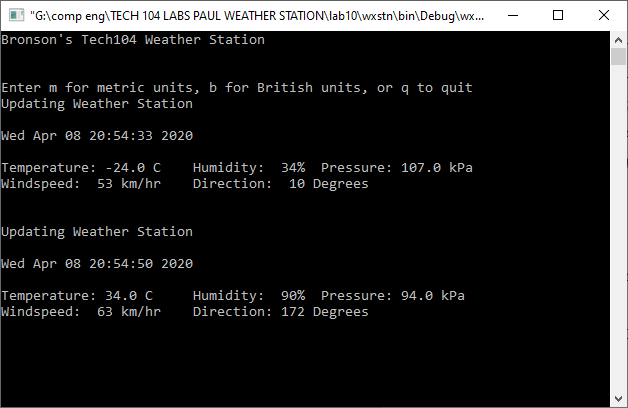
void msDelay(int ms);

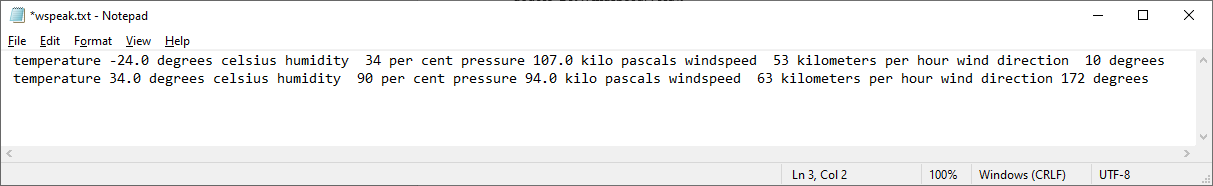
int LogWeatherData(char \* fname, time\_t wtime, int units, double wreads[WDATASZ]);

int SpeakWeatherData(int units, time\_t wtime, double wreads[WDATASZ]);

#endif // WX\_H\_INCLUDED

Output:





“temperature -24.0 degrees celsius humidity 34 per cent pressure 107.0 kilo pascals windspeed 53 kilometers per hour wind direction 10 degrees

temperature 34.0 degrees celsius humidity 90 per cent pressure 94.0 kilo pascals windspeed 63 kilometers per hour wind direction 172 degrees”