#include <cstdlib>

#include <iostream>

#include <unistd.h>

#include <string.h>

#include <fstream>

#include <stdio.h>

#include <unistd.h>

#include <string>

#include <limits>

#include <windows.h>

/\*

19th July 2012.

A program by Alex Avramenko.

Contact details: [aaavramenko@gmail.com](mailto:aaavramenko@gmail.com)

\*/

//Strange but have to stay the same or the program doesnt work.

#define ARRAY\_SIZE\_rows 100

#define ARRAY\_SIZE\_columns 1000

#define SENTENCE\_SIZE 10

using namespace std;

//osdafafasd

/\*This function will ask the user whether regarding the specified parameter

he wants to put it in him self, to select a file for it, or leave it at default.

The program will then run a loop to feed the array with the required data.\*/

void intro()

{

printf("\n Select\n input value (same for all runs) (v)\n jump\t\t\t\t (j)\n sequence spanning the runs \t(s) \n manual input \t\t\t(m)");

printf("\n a file with parameters\t (f)\n");

//clears one line for the getline() function later

//std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

}

void ask\_user( char a[][ARRAY\_SIZE\_columns][SENTENCE\_SIZE], int current\_row, int num\_of\_columns,int \*spaces)

{

int i; //counter

char User\_choice[1], useful\_string[SENTENCE\_SIZE]; //where the choice will be saved

char File\_name[10];

char User\_value[SENTENCE\_SIZE];

std::string line; //where the input will be saved

// The question will be reset in case the user made a mistace.

//Only needed on the first run.

//PRINTS THE CURRENT LINE

//compare the lines.

int comparison =0;

for(i=0;i<(num\_of\_columns-1);i++)

{

comparison+=(strcmp(a[current\_row][i],a[current\_row][i+1]));

}

if(comparison)

{

printf("[ ");

for(i=0;i<num\_of\_columns;i++)

{

printf("%s ",a[current\_row][i]);

}

printf("]");

}

else{

printf("[%s]",a[current\_row][1]);

}

//Sorry for the spagetti code but this was the way I found for the computer

//to read the enter as also a valid input. This can not be done with scanf()

//For some reason this getline here insists to read each time the first input as

//empty and hence I had to make one more value in the loop.

again:

line.clear();

std::getline( std::cin, line );

if( line.empty() )

{

return;

}

if( !line.empty())

{

strcpy(User\_choice, line.c\_str());

//SWITCH FOR THE CHOICE

switch (User\_choice[0])

{ //---------------------------------------------------VALUE OPTION

case 'v' :

printf("Value:");

scanf("%s", User\_value );

//printf("Number:%f", User\_number);

for(i=0;i<num\_of\_columns;i++)

{

strcpy(a[current\_row][i], User\_value);

}

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

break;

//-----------------------------------------------------FILE OPTION

case 'f':

printf("\n\nFile in same directory as program: ");

scanf("%s", File\_name);

strcat(File\_name, ".txt");

printf("%s", File\_name);

FILE \*file;

//system("start parameters.txt" );

//system("pause");

file=fopen(File\_name, "r");

if ( file == 0 )

{ //In case the file fails to open

printf( "Could not open file\n" );

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

goto again;

}

else

{ i=0;//counter

char c[1000][1000];

int j=0, k=0, amount=0;

//This loop will fill the array with the parameters

//from the file

printf("\n\nHow many data points from the file? If zero just reads. If negative will repeat the values read for rep times ");

scanf("%d", &amount);

printf("\n\nThe file reads:");

if(amount<0){

amount= amount\*(-1);

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

{

fscanf(file, " %s", User\_value);

printf("\t%s", User\_value);//optional

sprintf(useful\_string, "%s",User\_value);

strcpy(c[i], useful\_string);

}

printf("\n\nHow many reps of each point?");

int reps=0;

int position\_of\_c=0, j=0;

scanf("%d", &reps);

for (i=0; i< num\_of\_columns;i++)

{

position\_of\_c= (j==reps)? (position\_of\_c+1): position\_of\_c;

j= (j==reps) ? 0 : j;

strcpy(a[current\_row][i], c[position\_of\_c]);

j++;

}

}

else

{

if (amount!=0)

{

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

{

fscanf(file, " %s", User\_value);

printf("\t%s", User\_value);//optional

sprintf(useful\_string, "%s",User\_value);

strcpy(c[i], useful\_string);

}

for (i=0; i< num\_of\_columns;i++)

{

j= (j==amount) ? 0 : j;

strcpy(a[current\_row][i], c[j]);

j++;

}

}

else

{

i=0;

while (1)

{ if(i>=num\_of\_columns) break;

fscanf(file, " %s", User\_value);

printf("\t%s", User\_value);//optional

sprintf(useful\_string, "%s",User\_value);

strcpy(a[current\_row][i], useful\_string);

i++;

}

}

}

/\*

while (1)

{ if(i>=num\_of\_columns) break;

fscanf(file, " %s", User\_value);

printf("\t%s", User\_value);//optional

sprintf(useful\_string, "%s",User\_value);

strcpy(a[current\_row][i], useful\_string);

i++;

} \*/

fclose( file );//close the file

}

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

break;

//---------------------------------------------------BLANK SPACE OPTION

case 'j':

i=0;

printf("\nWhat jump would you like?:\n");

scanf("%d", &i );

\*spaces += i-1; //increases the loop value by that much.

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

break;

case 's':

float start\_value, end\_value, steps;

char Stepper[ARRAY\_SIZE\_columns][40];

printf("\nStart Value:");

scanf("%f", &start\_value );

printf("\nEnd Value:");

scanf("%f", &end\_value );

//printf("start value:%g, end value%g", start\_value, end\_value);

steps= ((end\_value-start\_value)>0?end\_value-start\_value:start\_value-end\_value) /(((float)num\_of\_columns)-(float)1);

//printf("\nfirst condition:%g\n",(end\_value-start\_value)>0?end\_value-start\_value:start\_value-end\_value );

//printf("steps:%g", steps);

for(i=0; i<num\_of\_columns; i++)

{

sprintf(Stepper[i], "%g", (start\_value>end\_value?end\_value:start\_value)+steps\*((float)i));

printf("\t%s", Stepper[i]);

strcpy(a[current\_row][i], Stepper[i]);

}

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

break;

case 'm':

char manual[40];

printf("\nPrint the input:");

for(i=0; i<num\_of\_columns; i++)

{

printf("\nRun%d:",(i+1));

scanf("%s", manual);

strcpy(a[current\_row][i], manual);

}

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

break;

default:

printf("You wrote: %s \nPlease write again v, j, s, m or f:", line.c\_str());

//std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

goto again;

}//END of switch

}//END OF if( !line.c\_str() );

}//END OF THE ASK FUNTION\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

////input: int int ARRAY\_SIZE\_rows + columns, array[][][],

//Print all inside the array. Remember that the number of columns, and sentance size

// has to be fixed but the number of rows can be flexy.

void Array\_print\_all(char a[][ARRAY\_SIZE\_columns][SENTENCE\_SIZE],

int num\_of\_rows, int num\_of\_columns)

{

int i, j, k;//define the counters

int comparison =0;

for(i=0; i< num\_of\_rows; i++)

{

printf("\n%d ", i+1);

comparison =0;

for(k=0;k<(num\_of\_columns-1);k++)

{

comparison+=(strcmp(a[i][k],a[i][k+1]));

}

if(comparison)

{

printf("[ ");

for(j=0; j< num\_of\_columns; j++)

{

printf("%s\t",a[i][j]);

}

printf("]");

}

else

{

printf("%s\t",a[i][0]);

}

}

}

void Save\_data\_file(char a[][ARRAY\_SIZE\_columns][SENTENCE\_SIZE],

int num\_of\_rows, int num\_of\_columns, char File\_name[])

{

int i, j, k;//define the counters

int comparison =0;

FILE \*file\_save;

file\_save = fopen(File\_name, "w+");

for(i=0; i< num\_of\_rows; i++)

{

comparison =0;

for(k=0;k<(num\_of\_columns-1);k++)

{

comparison+=(strcmp(a[i][k],a[i][k+1]));

}

if(comparison)

{

for(j=0; j< num\_of\_columns; j++)

{

fprintf(file\_save,"%s\t",a[i][j]);

}

fprintf(file\_save, "\n");

}

else

{

fprintf(file\_save,"%s\n",a[i][0]);

}

}

fclose(file\_save);

}

//Function to fill the array's row with a particular parameter.

//Used to create some default input.

void Array\_fill\_row(char value[], char a[][ARRAY\_SIZE\_columns][SENTENCE\_SIZE],

int current\_row, int num\_of\_columns)

{

int i;

for(i=0;i<num\_of\_columns;i++)

{

strcpy(a[current\_row][i], value);

}

}

//=============================================================================

//=============================================================================

void open\_file(char a[][ARRAY\_SIZE\_columns][SENTENCE\_SIZE],

int runs, int \*count\_of\_rows)

{

int i,j, current\_row;

tryagain:

string line;

char input\_file\_name[12];

cout << "\n\nPlease give the file name:";

scanf("%s", input\_file\_name);

strcat(input\_file\_name, ".txt");

printf("%s", input\_file\_name);

ifstream myfile (input\_file\_name);

if (myfile.is\_open())

{

current\_row = 0;

while ( myfile.good() )

{

getline (myfile,line);

//==============================================================================

// printf("\nx%sx\t", line.c\_str()); //This will print the read line.

char b[1000][1000];

char \* pch;

char str[1000];

strcpy(str,line.c\_str());

pch = strtok (str," \t");

i=0;

while (pch != NULL)

{

strcpy( b[i], pch);

//printf ("%d: %s\n pch:%s\n" ,i, b[i], pch);

pch = strtok(NULL, " \t");

i++;

}

//Now the program needs to decide how to fill the main array based on runs

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

{

strcpy(a[current\_row][j], b[j]);

//printf ("\nwritten1: i%d j%d b%s\t" ,i,j, b[j]);

}

if(i>0)//If i==0 then the b[i-1] reference will fail.

{

for(j=i;j<runs;j++)

{

strcpy(a[current\_row][j], b[i-1]);

//printf ("\nwritten2: i%d j%d b%s\t" ,i,j, b[i-1]);

}

}

//==============================================================================

current\_row++;

}

\*count\_of\_rows = 0;

\*count\_of\_rows += (current\_row-1);

myfile.close();

}

else

{

printf("\n\nUnable to open file");

printf("\n\nTry again? (y/n)");

char try\_again[1];

scanf("%s", try\_again);

if (strcmp(try\_again, "y")==0){goto tryagain;}

}

}//END OF OPENING A FILE

//==============================================================================

//--------------------------START OF MAIN PROGRAM-------------------------------

//==============================================================================

int main(int argc, char \*argv[])

{

chdir("RayReflec"); //Remove later when the program is ready

int i,j; //counters

int yes\_no; //Choice

char useful\_string[SENTENCE\_SIZE];

/\*Most bizzarly this line has to stay for the code to work!

I do not know why. \*/

char Reflec\_sentances[100][100]={""};

//==============================================================================

//Prints the introduction.

//cout << "\nThis is The Looper Program.\n Created by Alex Avramenko on 20th of July 2012.";

printf("\n\n\n\n\n\n\n\n\n");

printf(" ## ##### ##### ###### ####### ###### \n");

printf(" ## ## ## ## ## ## ## ## ## ## \n");

printf(" ## ## ## ## ## ## ## ## ## ## \n");

printf(" ## ## ## ## ## ###### ####### ###### \n");

printf(" ## ## ## ## ## ## ## ## ## \n");

printf(" ## ## ## ## ## ## ## ## ## \n");

printf(" ####### ##### ##### ## ####### ## ## \n");

printf("\n\n\n\n\n\n\n\n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

Sleep(1000);

printf("\n\nProgram created by Alex Avramenko on 20th of\n July 2012. Looper program was created to run\nRay/Reflec in loops. You should follow exactly\nwhat you type in Ray/Reflec and write it in a text\nfile. Put each new input on a new line in the text\nfile. There are two main ways to input\npermutations in to the program. Either manually\nwith the editor. There it will give you the option of\nuploading a file with the parameters. Or you could\nput the permutation in your original text file. For\nexample here is a sample text file for Reflec:\n\n\nNO\nNO\n1\n1\nGR\n620\n620\n0\nAu\n0\n19.3\n88.45\n86.08\n100\n-1\n3\n90\n10 12 14 16 18 20 22\n0.65\n11\n\nyes\n53\n7\n128\nyes\nno\nyes\n1\nyes\n1\n3\n10 12 14 16 18 20 22\nEXIT\n\nAs you can see each new line is the new line of\ninput in Reflec. And the lines with 10 12 22... are\nthe permutations of that line. Unfortunately your\ninput file MUST BE PRECISE and there is no\nroom for error.\nThe program will then save all the permutations of\nyour input file in Run1.txt, Run2.txt... These files it\nwill then feed in to Ray/Reflec. Ray/Reflec always\nsaves the files in the directory Files. Thus you\nshould remember to get Looper to permutate each\ntime the name of the file as other wise Ray/Reflec\nwill overwrite the file and you will end up with only\none file. Finally it will delete all the Run.txt files\nand just before it does so you have time to have a\nlook at them to see if you had any errors. If there\nis any error in the chain of program it will probably\nbe here where the Run.txt files have not been\ncreated accuratly and thus when fed to Ray/Reflec\ndo not produce the required results.\n\n\nGood Luck in using this program and I hope it will\nhelp to reduce the work load or better still increase\nproductivity. \n");

// system("PAUSE"); //pause until action

/\*Declaires the first array. More to come. The rows will hold the

number of runs the program will do. The colums are the columns for the input

file, and the sentence size is just to create a string array. This is

required in c code. \*/

char param[ARRAY\_SIZE\_rows][ARRAY\_SIZE\_columns][SENTENCE\_SIZE];

//===================CLear the array just in case.==============================

for(i=0; i< ARRAY\_SIZE\_rows; i++)

{

for(j=0; j< ARRAY\_SIZE\_columns; j++)

{

strcpy(param[i][j],"");

}

}

//==============================================================================

//=======READ THE FILE AND ASK FOR RUNS AND THEN LINES==========================

int lines=0, runs=0;

//ASK FOR RUNS

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\n\n\n\n\n\nHow many runs:");

scanf(" %d", &runs);

//printf("\nYou have given %druns\n", runs);

//OPEN FILE

open\_file(param, runs, &lines);//another importatnt function.

Array\_print\_all( param, lines, runs);

printf("\n\nThe program changed to %d lines and %d runs\n", lines, runs);

//CHANGE THE NUMBER OF LINES YES NO?

/\*

printf("\n\nWould you like to change the number of lines?: ");

printf("\n yes(1) no (0):");

scanf("%d", &yes\_no);

//IF YES

if(yes\_no==1)

{

printf("\nRemember that changing the number of lines allows to expand or trunkate the file.");

printf("how many lines:");

scanf(" %d", &lines);

} \*/

//==============================================================================

//====================ASK THE USER IF HE WANTS THE FILES SAVED IN SAME FOLDER===

/\*

int Files\_saved\_same\_folder;

printf("\n\nWould you like to save all the result ");

printf("files in to separate folders?:\n yes(1) no (0):");

scanf("%d", &Files\_saved\_same\_folder);

if (Files\_saved\_same\_folder==1)

{

printf("\n\nPlease remember that the files will be saved\nin new folder");

printf(" in Files1, Files2,... In order\nfor this to work you must delete");

printf(" the\npreviously created Files1, Files2,... in there.");

} \*/

// This code will appear later: if (strcmp(Files\_saved\_same\_folder, "y")==0)

//std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

//==============================================================================

//================START OF ASKING THE USER FOR SOME INPUT=======================

//Extra code for potential future.

/\*Array\_fill\_row("hello", param, 0, ARRAY\_SIZE\_columns); \*/

//This will clear the buffer to make sure that the getline() doesn't skip.

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

//=======================WOULD YOU LIKE TO EDIT DATA?===========================

printf("\n\nWould you like to edit the data?: ");

printf("\n\n yes(1) no (0):");

scanf("%d", &yes\_no);

//IF YES

if(yes\_no==1)

{

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

intro(); //This will give the intro on the edit.

reedit:

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

{

printf("\nline: %d \n",(i+1));

ask\_user( param, i, runs, &i);

}

Array\_print\_all( param, lines, runs);

printf("\n\nWould you like to edit the data further?: ");

printf("\n\n yes(1) no (0):");

scanf("%d", &yes\_no);

if(yes\_no==1)

{

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

goto reedit;

}

{//Start of Yes no

int choice;

char Proposed\_file\_name[12];

cout << "\n\nWould you like to save your edited file\n yes(1) no (0):";

cin >> choice;

yes\_no\_again:

switch (std::tolower(choice)) {

case 1:

cout << "\n\nFile name:";

cin >> Proposed\_file\_name;

strcat(Proposed\_file\_name, ".txt");

//printf("%s", Proposed\_file\_name);

Save\_data\_file(param, lines, runs, Proposed\_file\_name);

cout << "\n\nYour file has been saved";

break;

case 0:

break;

default:

std::cout << "\n\n Please print 0 or 1";

goto yes\_no\_again;

break; }//End of Yes no

}

}

//==============================================================================

//===Creates all the input files to be used to feed the Reflec.exe program======

char filename[runs][15];

FILE \*input\_file;

//name all the files

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

{

strcpy(filename[i], "Run");

// strcpy(useful\_string, "");

sprintf(useful\_string, "%d" , (i+1));

//printf("%s", useful\_string);

strcat(filename[i], useful\_string);

strcat(filename[i], ".txt");

//printf("\n%s", filename[i]);

}

printf("\n\nThe run files have been created");

//==============================================================================

/\* for(i=0; i<runs; i++)

{

printf("\n%s",filename[i]);

} \*/

//==========================Uploads the array to the input files---------------

// input: int ARRAY\_SIZE\_columns, char name of file, int ARRAY\_SIZE\_rows

for(j=0,i=0; j<runs;j++)

{

//printf("i=%d j=%d %s\t %s\n",i,j,param[i][j],filename[j]);

input\_file=fopen(filename[j], "w+");

//if(j%5==0){system("pause");} //To be removed

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

{

//printf("bbbi=%dj=%d %s\t",i,j,param[i][j]);

fprintf(input\_file,"%s\n",param[i][j]);

//WRITE MORE CODE HERE TO UPLOAD ADDITIONAL ARRAYS TO THE

// FILE. Even empty ones using the following code:

//fprintf(input\_file,"%s\n",param[i][j]);

}

fclose(input\_file);

}

//==============================================================================

//ask the user on what he wants to do with those files.

char User\_choice[1];

char system\_feed\_name[SENTENCE\_SIZE];

char new\_directory\_name[SENTENCE\_SIZE];

printf("\n\nDo you want to run those files in:\n Ray (y), Reflec(f) or quit(q):");

scanf("%s", &User\_choice[0]);

switch (User\_choice[0])

{

case 'y' :

//Perform the run and coppy the folder away.

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

{

//mkdir("Files");

//printf("a");

//Run ray!

strcpy(system\_feed\_name, "Ray.exe < ");

strcat(system\_feed\_name, filename[i]);

//printf("\n%s\n", system\_feed\_name);

//system("dir");

system(system\_feed\_name);

//printf("\nb\n");

/\*Move the folder

sprintf(useful\_string, "%d" , (i+1));

//printf("%s", useful\_string);

mkdir("new folder");

strcpy(new\_directory\_name, "new folder/Files");

strcat(new\_directory\_name, useful\_string);

//printf("\n %s\n", new\_directory\_name);

if (Files\_saved\_same\_folder==1)

{

rename("Files", new\_directory\_name);

}

mkdir("Files");

//rename("Files2", "new folder/Files2"); \*/

}

break;

case 'f' :

//All the same as above except that now its for Reflec

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

{

//mkdir("Files");

//printf("a");

//Run ray!

strcpy(system\_feed\_name, "Reflec.exe < ");

strcat(system\_feed\_name, filename[i]);

//printf("\n%s\n", system\_feed\_name);

//system("dir");

system(system\_feed\_name);

//printf("\nb\n");

/\*Move the folder

sprintf(useful\_string, "%d" , (i+1));

strcpy(new\_directory\_name, "new folder/Files");

strcat(new\_directory\_name, useful\_string);

//printf("\n %s\n", new\_directory\_name);

if (Files\_saved\_same\_folder==1)

{

rename("Files", new\_directory\_name);

}

mkdir("Files"); \*/

}

break;

case 'q' :

break;

}

printf("\n\nCompleted Program\n\n");

printf("\n\n Make a copy of your run files if you wnat,\n or now they will be deleted.\n");

system("PAUSE");

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

{

remove(filename[i]);

}

return EXIT\_SUCCESS;

}

//==============================================================================

//----------------------------END OF MAIN PROGRAM-------------------------------

//==============================================================================

//relics of code for potential future use.

//This one prints everything inside the array.

/\* for(i=0; i< ARRAY\_SIZE\_rows; i++)

{ printf("\n");

for(j=0; j< ARRAY\_SIZE\_columns; j++)

{

printf("%s\t",param[i][j]);

}

} \*/

//This was a potential hope for me to run the program from inside c and get feed

//directly from. I didn't manage for it to work.

/\*void exec() {

FILE\* pipe = popen("Reflec.exe 2>&1", "rb");

char buffer[500]={""};

printf("a");

char User\_input[10];

while(!feof(pipe)) {

fgets(buffer, 2, pipe);

printf("%s", buffer);

// printf("%e", ftell(pipe));

//if(strcmp(buffer,":")==0){

// scanf("%s", User\_input);

// printf("registered%s", User\_input);

// }

//printf("" );

}

pclose(pipe);

}

\*/

//Some relic code to run the Ray.exe program with the input files. To be redone.

/\* system( "dir" );

chdir("RayReflec"); // changes the directory

mkdir("Files");

// system("cd RayReflec");

//system("dir");

system("Ray.exe < INPUT.txt");

//yey this creates a new folder

rename("Files", "Files2"); //successful

rename("Files2", "new folder/Files2"); //the file is successfully moved!!!

// system( "start INPUT.txt" );

\*/