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

// Name : ProductiveApplication1.cpp

// Author : Libby&Sandhya

// Version :

// Copyright : Your copyright notice

// Description : Hello World in C++, Ansi-style

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

**using** **namespace** std;

**#include** <iostream>

**#include** <string>

**#include** <bits/stdc++.h>

**#ifdef** WINDOWS

**#include** <direct.h>

**#define** GetCurrentDir \_getcwd

**#else**

**#include** <unistd.h>

**#define** GetCurrentDir getcwd

**#endif**

**#include** "HelperHeader/FileProcessing.h"

**#include** "HelperHeader/Configuration.h"

**#include** "HelperHeader/TextTable.h"

**#include** "HelperHeader/MonthlyBudgetHelper.h"

**#include** "Header/Topic.h"

**#include** "Header/Habit.h"

**static** FileProcessing fileProcessing;

string **get\_current\_dir**() {

**char** buff[FILENAME\_MAX]; //create string buffer to hold path

GetCurrentDir( buff, FILENAME\_MAX );

string current\_working\_dir(buff);

**return** current\_working\_dir;

}

vector<Topic> **loadTopics**(){

vector<Topic> topics;

Topic DrinkWater("Drink Water", "Water Keeps you hydrated and flushes out toxins");

Topic EatFandV("Eat Fruits and Vegetables", "Natural Source of essential nutrients and fibers");

Topic Exercises("Do Morning Exercises", "Do charge your battery");

Topic Walk("Go For a Walk", "Walking strengthens your bones and improves your mood");

Topic Read("Read", "Reading newspaper or books broadens the mind,so pick a topic now");

Topic Sleep("Sleep for 8 hours atleast", "Your body will be grateful");

Topic MakeTime("Make time for yourself", "Stop the daily rush and listen carefully");

Topic Meditate("Meditate", "Find inner peace");

Topic SetGoals("Set Goals", "Stay Motivated and focused");

Topic SpendTime("Spend time with family", "Stay connected");

//Main Habits

Topic mustHaveHabits("Must have habits", "Try to follow");

vector<Topic> subTopics;

subTopics.push\_back(DrinkWater);

subTopics.push\_back(EatFandV);

subTopics.push\_back(Exercises);

subTopics.push\_back(Walk);

subTopics.push\_back(Read);

subTopics.push\_back(Sleep);

subTopics.push\_back(MakeTime);

subTopics.push\_back(Meditate);

subTopics.push\_back(SetGoals);

subTopics.push\_back(SpendTime);

mustHaveHabits.setSubTopics(subTopics);

Topic morningRoutine("Morning routine", "Try to follow");

vector<Topic> subTopics1;

subTopics1.push\_back(DrinkWater);

subTopics1.push\_back(EatFandV);

subTopics1.push\_back(Exercises);

subTopics1.push\_back(Walk);

morningRoutine.setSubTopics(subTopics1);

topics.push\_back(mustHaveHabits);

topics.push\_back(morningRoutine);

**return** topics;

}

Scheduler **getShedule**(Topic topic){

Scheduler shedule(topic.getTitle());

cout<<**endl**<<"Select the frequency: ";

cout<<**endl**<<"1.Daily";

cout<<**endl**<<"2.Weekly";

cout<<**endl**<<"3.Monthly";

cout<<**endl**<<"Enter the frequency you want to add: ? ";

**int** number, number1, number2;

cin>>number;

**switch**(number){

**case** 1:{

shedule.setFrequency("Daily");

**break**;}

**case** 2:{

shedule.setFrequency("Weekly");

cout<<**endl**<<"How many Times a week (Enter1-7): ";

cin>>number1;

vector<string> duration;

duration.push\_back(number1+"");

shedule.setDuration(duration);

**break**;}

**case** 3:{

shedule.setFrequency("Monthly");

cout<<**endl**<<"Select the time of the month: ";

cout<<**endl**<<"1.Start";

cout<<**endl**<<"2.Middle";

cout<<**endl**<<"3.End";

cout<<**endl**<<"4.Whole";

vector<string> duration1;

cout<<**endl**<<"U can enter multiple time so tell me how many time you want to select ? ";

cin>>number1;

cout<<**endl**<<"Enter the time of the month you want to add: ? : ";

number1 = number1>4? 4 : number1;

**for**(**int** i=0;i<number1;i++){

cin>>number2;

**if**(number2==1) duration1.push\_back("Start");

**else** **if**(number2==2) duration1.push\_back("Mid");

**else** **if**(number2==3) duration1.push\_back("End");

**else** duration1.push\_back("Whole");

}

shedule.setDuration(duration1);

**break**;}

deafult:{

shedule.setFrequency("Monthly");

vector<string> duration2;duration2.push\_back("Whole");

shedule.setDuration(duration2);

**break**;}

}

cout<<**endl**<<"Select the time of the Day: ";

cout<<**endl**<<"1.Morning";

cout<<**endl**<<"2.AfterNoon";

cout<<**endl**<<"3.Eveing";

cout<<**endl**<<"4.Night";

cout<<**endl**<<"Which Time of the day you prefer please enter? ";

cin>>number;

**if**(number==1) shedule.setTimeOfDay("Morning");

**else** **if**(number==2) shedule.setTimeOfDay("AfterNoon");

**else** **if**(number) shedule.setTimeOfDay("Eveing");

**else** shedule.setTimeOfDay("Night");

**return** shedule;

}

**void** **process1Habit**(Configuration& configuration, string habitFileName){

cout<<**endl**<<**endl**;

vector<Topic> topics = configuration.getDisplayTopics();

cout<<**endl**<<"0.Quit";

**for**(**int** i=0;i<topics.size();i++){

cout<<**endl**<<(i+1)<<"."<<topics[i].getTitle();

}

**while**(1){

cout<<**endl**<<"Enter the topic you want to add: ? ";

**int** topicNumber;

cin>>topicNumber;

**if**(topicNumber>0 && topicNumber<=topics.size()){

cout<<**endl**<<"You have selected number: "<<topicNumber;

cout<<**endl**<<"Title : "<<topics[topicNumber-1].getTitle();

cout<<**endl**<<"Description : "<<topics[topicNumber-1].getDescription();

cout<<**endl**<<"You can choose between the following subtopics : ";

vector<Topic> subtopics = topics[topicNumber-1].getSubTopics();

cout<<**endl**<<"0.Quit";

**for**(**int** i=0;i<subtopics.size();i++){

cout<<**endl**<<(i+1)<<"."<<subtopics[i].getTitle();

}

cout<<**endl**<<"Enter the sub topic you want to add : ? ";

cin>>topicNumber;

**if**(topicNumber>0 && topicNumber<=subtopics.size()){

cout<<**endl**<<"SubTopic is added";

Scheduler scheduler = getShedule(subtopics[topicNumber-1]);

fileProcessing.WriteToFile(habitFileName,scheduler);

**break**;

}**else** **if**(topicNumber==0){

cout<<**endl**<<"Coming out.....";

**break**;

}**else**{

cout<<**endl**<<"Invalid selection";

}

}

**else** **if**(topicNumber==0){

cout<<**endl**<<"Coming out.....";

**break**;

}**else**{

cout<<**endl**<<"Invalid selection";

}

}

cout<<**endl**<<**endl**;

}

**void** **display2Habits**(string fileName){

vector<string> content = fileProcessing.ReadFileLineByLine(fileName);

TextTable t( '-', '|', '+' );

t.add( "Habit Title" );

t.add( "Frequency" );

t.add( "Time of the Day" );

t.add( "Duration" );

t.endOfRow();

**for**(**int** i=0;i<content.size();i++){

vector <string> tokens;

// stringstream class check1

stringstream check1(content[i]);

string intermediate;

// Tokenizing w.r.t. space ' '

**while**(**getline**(check1, intermediate, '\t'))

{

tokens.push\_back(intermediate);

}

**for**(**int** i=0;i<3;i++){

t.add(tokens[i]);

}

string durationString;

**for**(**int** i=3;i<tokens.size();i++){

durationString = durationString + tokens[i] + " / ";

}

t.add(durationString);

t.endOfRow();

}

t.setAlignment( 2, TextTable::Alignment::*RIGHT* );

std::cout << t;

}

**void** **process3ExpenseManager**(){

MonthlyBudgetHelper helper;

MonthlyBudget spentBudget;

MonthlyBudget fixedBudget = { 5000.0, 1500.0, 650.0, 500.0, 2500.0, 300.0, 1000.0, 1500.0, 750.0, 500.0, 14200.0 };

helper.getBudget(spentBudget);

helper.calcTotal(spentBudget);

helper.displayBudget(spentBudget, fixedBudget);

}

**void** **process4Quotes**(Configuration& configuration){

**int** readPointer = configuration.getReadPointerQuote();

cout<<**endl**<<**endl**<<configuration.getQuotes()[readPointer];

**if**(readPointer<75)

readPointer++;

**else**

readPointer = 0;

configuration.setReadPointerQuote(readPointer);

cout<<configuration.getReadPointerQuote();

cout<<**endl**<<**endl**;

}

**void** **updateConfiguration**(Configuration& configuration){

string configFileName =get\_current\_dir() + "\\" + "src"+ "\\" + "Files"+ "\\" +"configuration" + ".txt";

vector<string> content ;

content.push\_back(configuration.getReadPointerQuote()+"");

fileProcessing.DeleteAndWriteToFile(configFileName,content);

}

**void** **runMenu**(Configuration& configuration){

string habitFileName = get\_current\_dir() + "\\" + "src"+ "\\" + "Files"+ "\\" +"habits" + ".txt";

cout<<"\t!!!\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*!!!" << **endl**; // prints !!!Hello World!!!

cout<<"\t\t\tHandy Productive App"<< **endl** ;

cout<<"\t!!!\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*!!!" << **endl** << **endl**;

**int** option = 9;

**while**(1){

cout<<"\t\t ---Main Menu---"<< **endl**;

cout<<"\t\t 1.Add New Habit Organizer"<< **endl**;

cout<<"\t\t 2.View your Habits"<< **endl**;

cout<<"\t\t 3.Expense Manager"<< **endl**;

cout<<"\t\t 4.Quotes of the Day"<< **endl**;

cout<<"\t\t 5. Quit"<< **endl**;

cout<<**endl**<<**endl**<<"\t\tEnter your option : ";

cin>>option;

**switch**(option){

**case** 1:

process1Habit(configuration,habitFileName);

**break**;

**case** 2:

display2Habits(habitFileName);

**break**;

**case** 3:

process3ExpenseManager();

**break**;

**case** 4:

process4Quotes(configuration);

**break**;

**case** 5:

cout<<"\n\n\t\tThank you see you soon!!! ";

**getchar**();

**exit**(0);

**break**;

**default**:

cout<<"\n\n\t\tWrong selection!!";

cout<<"\n\n\t\tThank you see you soon!!! ";

**getchar**();

**exit**(0);

**break**;

}

}

}

Configuration **loadConfiguration**(){

string currentDir = get\_current\_dir();

string configFileName =currentDir + "\\" + "src"+ "\\" + "Files"+ "\\" +"configuration" + ".txt";

vector<string> configContent = fileProcessing.ReadFileLineByLine(configFileName);

//1.Habit

vector<Topic> topics = loadTopics();

//7.Quotes

string quotesFileName = currentDir + "\\" + "src"+ "\\" + "Files"+ "\\" +"quotes" + ".txt";

vector<string> quotes = fileProcessing.ReadFileLineByLine(quotesFileName);

//8.Jokes

//string jokesFileName = currentDir +"\\" + "src"+ "\\" + "Files"+ "\\" +"configuration" + ".txt";

//vector<string> jokes = fileProcessing.ReadFileLineByLine(jokesFileName);

Configuration config(stoi(configContent[0]));

config.setDisplayTopics(topics);

config.setQuotes(quotes);

**return** config;

}

**int** **main**() {

//Load Configs

Configuration configuration = loadConfiguration();

runMenu(configuration);

updateConfiguration(configuration);

**return** 0;

}

/\*

\* Habit.h

\*/

**#include**<string>

**#include**<vector>

**using** **namespace** std;

**#include** "../Header/Scheduler.h"

**class** Habit

{

**private**:

string topic;

string subTopic;

Scheduler schedule;

**public**:

**Habit**(string,string,Scheduler);

**~Habit**(**void**){};

string **getTopic**();

string **getSubTopic**();

Scheduler **getScheduler**();

};

/\*

\* MonthlyBudget.h

\*/

**struct** MonthlyBudget

{

**double** housing; /\* Amount spent for housing \*/

**double** utilities; /\* Amount spent for utilities \*/

**double** householdExp; /\* Amount spent for household expenses \*/

**double** transport; /\* Amount spent for transportation \*/

**double** food; /\* Amount spent for food \*/

**double** medical; /\* Amount spent for medical treatment \*/

**double** insurance; /\* Amount spent for insurance \*/

**double** entertainment; /\* Amount spent for entertainment \*/

**double** clothing; /\* Amount spent for clothing \*/

**double** miscellaneous; /\* Amount spent for miscellaneous items \*/

**double** total; /\* Total amount spent \*/

};

**#pragma** once

**#include** "../Header/MonthlyBudget.h"

**class** MonthlyBudgetHelper{

**public**:

**MonthlyBudgetHelper**(){};

**~MonthlyBudgetHelper**(**void**){};

**void** **getBudget**(MonthlyBudget &);

**void** **calcTotal**(MonthlyBudget &);

**void** **calcBudget**(**const** **double**, **const** **double**);

**void** **displayBudget**(**const** MonthlyBudget, **const** MonthlyBudget);

};

**#pragma** once

**#include**<string>

**#include**<vector>

**using** **namespace** std;

**class** Scheduler

{

**private**:

string title;

string frequency;

vector<string> duration;

string timeOfDay;

string reminderTime;

**public**:

**Scheduler**(string title);

**~Scheduler**(**void**){};

**void** **setFrequency**(string);

**void** **setDuration**(vector<string>);

**void** **setTimeOfDay**(string);

**void** **setReminderTime**(string);

string **getTitle**();

string **getFrequency**();

vector<string> **getDuration**();

string **getTimeOfDay**();

string **getReminderTime**();

};

**#pragma** once

**#include**<string>

**#include**<vector>

**using** **namespace** std;

**class** Topic

{

**private**:

string title;

string description;

vector<Topic> subTopics;

**public**:

**Topic**(){};

**Topic**(string,string);

**~Topic**(**void**){};

**void** **setSubTopics**(vector<Topic>);

string **getTitle**();

string **getDescription**();

vector<Topic> **getSubTopics**();

};

**#include** "../Header/DiaryRecord.h"

DiaryRecord::**DiaryRecord**(**int** id, string title, string place, string note, string time):

id(id),

title(title),

place(place),

note(note),

time(time){}

**int** DiaryRecord::**getId**(){

**return** id;

}

string DiaryRecord::**getTitle**(){

**return** title;

}

string DiaryRecord::**getPlace**(){

**return** place;

}

string DiaryRecord::**getNote**(){

**return** note;

}

string DiaryRecord::**getTime**(){

**return** time;

}

**void** DiaryRecord::**setTitle**(string newTitle){

title = newTitle;

}

**void** DiaryRecord::**setPlace**(string newPlace){

place = newPlace;

}

**void** DiaryRecord::**setNote**(string newNote){

note = newNote;

}

**void** DiaryRecord::**setTime**(string newTime){

time = newTime;

}