System Manual:-
/
****
PROGRAM : lunar_lander.cpp
Coder : Mosfiqur Rahman (mr986@drexel.edu)
Last Modified : 11th March, 2016.
**/
PROGRAM PURPOSE:

The is a small game named Lunar Lander, which is based on a variety of similarly themed games inspired by the Apollo 11 space mission of July, 1969 in which US astronauts first landed on the Moon. After running the program, it'll engage the user in a session that plays the game, updates the values of height and velocity at each turn and stops when the spacecraft reaches the surface (aka touchdown). At this point, a final analysis is printed.

includes:

iostream
cmath
string
fstream
algorithm

namespaces:

std

Functions:

int main()
void reportStatus(ostream &os, double time, double height, double velocity, double fuel, string name)
void updateStatus(double &velocity, double burnAmount, double &fuelRemaining, double &height)
void introduction(istream &is, ostream &os, string target,string replacement)
void touchdown(double &elapsedTime, double &velocity, double&burnAmount, double &height)
void finalAnalysis(ostream &os, double velocity)

int main()

- Declares a double type variable named height, and initializes the value to 1000 feet
- Declares a double type variable named velocity, and initializes the value to 50feet/sec
- Declares a double type variable named time, and initializes the value to 0sec
- Declares a double type variable named fuel, and initializes the value to 150 units
- Declares a double type variable named burn_fuel
- Declares a string type variable named name, and initializes the value to "APOLLO"
- Declares a string type variable named filename
- Declares a string type variable named check
- Declares a string type variable named target, and initializes the value to "$SPACECRAFT"
- Declares a ifstraem object is and sets the file name to "input.txt"
- Asks the user to file name where they would like to log their session

- Declares a ofstraem object os and sets the file name as directed by the user
- Asks the user whether they wants to see the instruction or not
- Checks whether the value of check is equal to Y or y
- Calls the function introduction(istream &is, ostream &os, string target,string replacement), and passes the value
- Calls the function reportStatus(ostream &os, double time, double height, double velocity, double fuel, string name) and passes the value
- Increments the value of time
- Runs a while loop to get the amount of fuel that the user would like to burn
- Initializes the value of the variable burn_fuel
- Calls the function updateStatus(double &velocity, double burnAmount, double &fuelRemaining, double &height) and passes the value
- Calls the function reportStatus(ostream &os, double time, double height, double velocity, double fuel, string name) and passes the value
- Calls the function touchdown(double &elapsedTime, double &velocity, double&burnAmount, double &height) and passes the value
- Shows final output for touchdown time
- Shows final output for landing velocity
- Shows final output for remaining fuel
- Calls the function finalAnalysis(ostream &os, double velocity) and passes the value
- Sets output to the output file as directed by the user
- Closes the output file

void reportStatus(ostream &os, double time, double height, double velocity, double fuel, string name)

*/****

This function outputs informations related with the time, height, speed, and fuel repeatedly

@param &os - The ofstream object
@param time - The elapsed time
@param height - The height from the ground
@param velocity - The velocity of the falling spaceship
@param fuel - The amount of remaining fuel
@param name - The name of the spaceship, i.e. APOLLO

****/*

- Shows the elapsed time as output
- Shows the certain height from the ground
- Shows the speed of the spaceship
- Shows the amount of remaining fuel
- Sets output to the output file as directed by the user

void updateStatus(double &velocity, double burnAmount, double &fuelRemaining, double &height)

*/****

This function uses the current values of velocity, burnAmount, fuelRemaining and height to compute the values one second later

@param &height - The height from the ground
@param &velocity - The velocity of the falling spaceship
@param &fuelRemaining - The amount of remaining fuel
@param burnAmount - The amount of fuel that is going to be burnt

****/*

- Declares a double type variable named old_velocity, and initializes the value to velocity
- Declares a double type variable named old_height, and initializes the value to height
- Declares a double type variable named old_fuel, and initializes the value to fuelRemaining
- Computes the value of the variable fuelRemaining
- Computes the value of the variable velocity
- Computes the value of the variable height

void introduction(istream &is, ostream &os, string target,string replacement)

*/****

This function creates a set of instructions replacing all instances of some string target from the input stream with the string specified by the replacement parameter for the output stream

@param &is - The ifstream object
@param &os - The ofstream object
@param target - The specified string, which is going to be replaced to
@param replacement - The specified string, which is going to be replaced with

****/*

- Declares a string variable named line
- Declares a string variable named replacement_final
- Runs awhile loop to find the targeted string from the file
- Declares a integer type variable named index, and initializes the value to the starting point of the targeted string
- Runs a while loop to replace the targeted string
- Initializes the value of of the string line to replacement_final
- Shows the output of new with the new value of the string
- Sets output to the output file as directed by the user

void touchdown(double &elapsedTime, double &velocity, double&burnAmount, double &height)

*/****

This function does some corrections at the point of touchdown in order to get accurate final values of height and velocity

@param &height - The height from the ground
@param &velocity - The velocity of the falling spaceship
@param &elapsedTime - The amount of time that has elapsed
@param &burnAmount - The amount of fuel that is going to be burnt

****/*

- Declares a double type variable named frac
- Checks whether the value of burnAmount is equal to 5
- Computes the value of the variable frac = height / velocity
- Or, Computes the value of the variable frac = (sqrt((pow(velocity, 2)) + (height*(10 - 2*burnAmount))) - velocity)/(5- burnAmount)
- Computes the value of the variable elapsedTime
- Computes the value of the variable velocity
- Computes the value of the variable height

void finalAnalysis(ostream &os, double velocity)

*/****

This function reports the damage based on the final velocity

@param &os - The ofstream object
@param velocity - The velocity of the falling spaceship

****/*

- Checks and compares the value of velocity through different logic statements
- Shows the report according to the condition
- Sets output to the output file as directed by the user