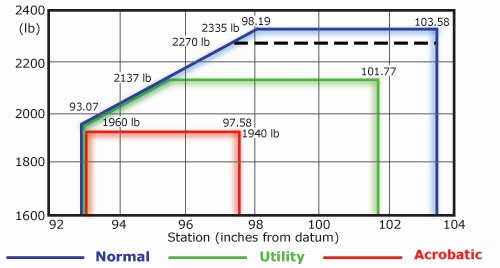
**PROJECT 1**

**SMALL AIRCRAFT CG CALCULATOR**

***For use of modifying permanent equipment records (TCDS)***

***per FAA-H-8083-1A***

**CSC-5: 42450**

**April 28th, 2014**

**Adams, Jonathan**

**TABLE OF CONTENTS**

**Page 3: Introduction/Summary**

**Pages 4-5: Defining technical terms & their variables**

**Page 6: Other Variables/Future Features**

**Page 7: Flow Chart**

**Page 8: Code**

**Introduction:**

This program is to assist in performing the computations for different tasks commonly encountered in aircraft maintenance. I wanted to program this to one day provide myself with something I can actually use in my line of work. I wanted to have my program knowledge and job knowledge complement each other, given I put time into both. This way I can effectively improve both skills.

My end goal is to become an engineer, however, I am hoping as I go down that road I can use my new skills to create actual useful products, if not for the entire industry that I will be leaving, at least for the people I worked with and the specific tasks they tackle. But overall it is the entrepreneur spirit that lead me to want to program this sort of project versus a game.

**Summary:**

Right now the only thing programmed is a CG calculator that can be used to make the basic calculations when changing permanent equipment on an aircraft Type Certificate Data Sheet. It is this document that the input numbers for limitations on fuel, cargo and passengers is derived. So, it is only to be modified when doing a major alteration to the aircraft, such as changing an aircraft starter or adding a piece of avionics equipment.

It is not to be used for fuel or cargo loading CG. In the future I would like to add this, but it will involve adding a lot of data, because those calculations are based on charts that are derived from the permanent equipment certificate as well as other manufacturer data.

I will also be adding future calculators that mechanics should use on a daily basis, such as bending sheet metal equations or torque conversions when using a torque wrench with extensions or adapters.

**Important Terms:**

The CG calculator will be confusing to someone not familiar with basic aircraft weight and balance. So I will define a few terms so anyone could use the program to get a feel to how CG works on small aircraft.

**Center of Gravity:**

**Variable: float acftCG**

The point in which the weight of one side of a point is equal to the weight on the other side of a point. It is where the aircraft "weighs" the most.

Most of the time pilots and maintainers are generally focused on and worried about the CG point about the lateral axis, which means the point of balance of the forward and aft dimension. This is the point, if the aircraft were suspended from, where the nose and tail of the aircraft will neither dip or rise. It is important that the CG is near the center of lift. All aircraft have a specified range of allowable CG limits.

NOTE: These limits have not been included in this code, to include the C-172M profile.

Whenever any weight on the aircraft shifts aft or forward, the aircraft CG changes.

**Basic Empty Weight:**

**Variable: float acftW**

Weight of the aircraft with zero fuel, zero passengers, zero cargo. It does include equipment permanently installed on the aircraft. It is where fuel, cargo and other loading charts are derived from. So, it is not done often or lightly.

**Aircraft Moment:**

**Variable: float acftMom**

Aircraft moment is a value of force applied to an aircraft at a given station number (aircraft is divided in inches) with a given weight at that station #. Thus, whenever weight is added, removed or shifted, the aircraft moment also changes.

**Datum point:**

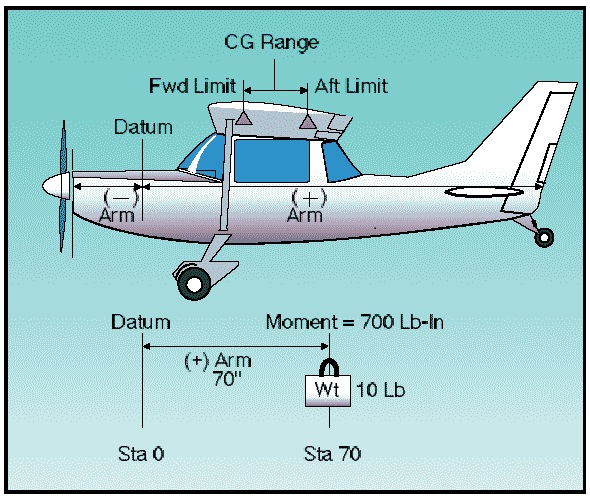
An imaginary vertical plane that all measurements of horizontal distances/ARM are taken from, for weight and balance. The reference datum is set by the aircraft manufacturer and is set a "0". On small aircraft, it is usually located at the firewall between the cockpit and engine bay. However, it can be located somewhere not even on the aircraft.

NOTE: For this program, it is assumed that the mechanic understands this and it is a requirement that he knows where the datum point is, and how far away the equipment they are changing is, from the Datum. The program will ask the user whether or not they are operating forward or aft of the Datum.

**Arm:**

**Variable float armOld & armNew**

Aircraft ARM is the distance of a point on the aircraft to the datum. Arm multiplied by weight gives the moment for that piece of equipment at that station on the aircraft.



**Other variables:**

**Menus:**

unsigned short choose (main menu) menu1 (switch 1 menu),

char exit (exit main menu switch),

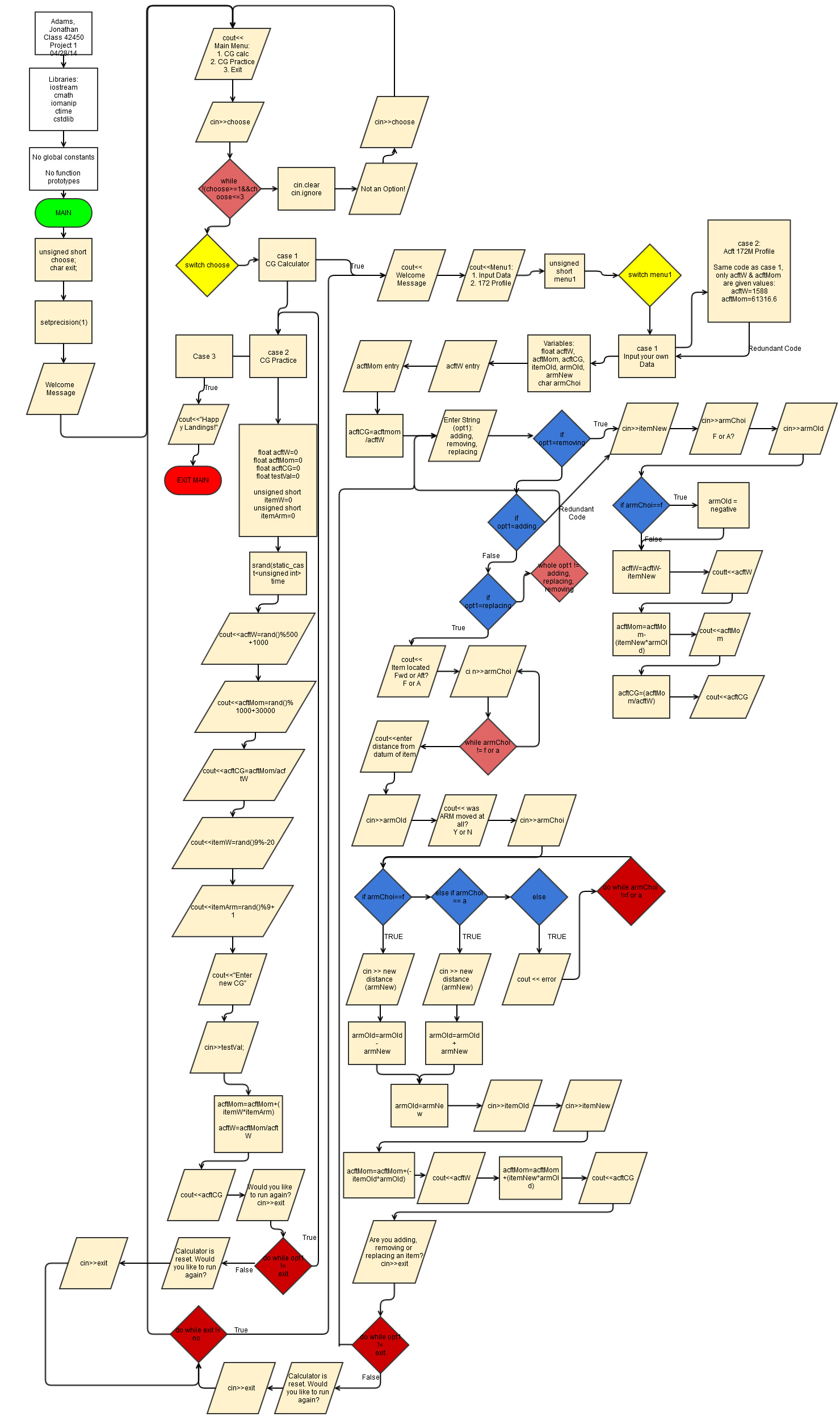
**True/False Logic:**

char armChoi (Yes/No or Fwd/Aft entry)

**Future features/Things I learned:**

I wished I would have started this program after I began learning functions. I will definitely be starting from scratch to clean up the code by using functions and a more efficient menu system.

A feature I really want to add is CG limits to the aircraft, so the user will know whether or not their new CG is allowable.



Pseudo Code:

Main Menu: 1. CG Calculator

2. CG Practice

3. Exit

1. CG Calculator:

Calculator Menu: 1. Enter Your Own Data

2. Cessna 172M Profile

1. Ask user for acftW & acftM:

Ask the user if they would like to add, remove or replace an item

if remove:

User is prompted to enter weight of item being removed.

User is prompted to enter moment of item being removed.

User is asked if ARM is forward or aft of datum.

Calculate and display new CG

if adding:

User is prompted to enter weight of item being removed.

User is prompted to enter moment of item being removed.

User is asked if ARM is forward or aft of datum.

Calculate and display new CG

if replacing:

User is prompted on ARM location; FWD or AFT of Datum?

User is prompted on whether or not ARM was adjusted slightly?

User is prompted on removed item's weight

User is prompted on new item's weight.

Calculates new moment

Calculates and displays new CG

2. Cessna 172 Profile is the same as Input Your Own Data, but sets a value to original aircraft weight and aircraft moment.

2. CG Practice:

Set random seed

Computer generates aircraft weight, moment, CG, a new item's weight and the new item's moment.

User is prompted to enter new CG

Display new CG

3. Exit

Exit program

**Code:**

#include <iostream>

#include <cmath>

#include <iomanip>

#include <ctime>

#include <cstdlib>

using namespace std;

//Global Constants

//Function Prototypes

//Execution

int main(int argc, char\*\* argv) {

//Declare Variables

unsigned short choose;

char exit;

//-- 1 Decimal point is sufficient for all calculations

//in this program

cout<<fixed<<setprecision(1)<<showpoint;

//Welcome statement to not be looped

cout<<"Welcome to the Aerospace and Powerplant Maintenance Helper."<<endl;

//Prompt user for number of problem to execute

cout<<endl;

cout<<endl;

do { //MENU DO LOOP BEGIN

cout<<"Choose from the following list"<<endl;

cout<<"1. Re-Calculate Longitudinal CG for new component"<<endl;

cout<<"2. Longitudinal CG Practice"<<endl;

cout<<"3. Exit Program - All"<<endl;

cin>>choose;

//Catch invalid input choose

while (!(choose>=1 && choose<=4)) {

cin.clear();

cin.ignore();

cout<<"Not an option! Please re-enter: "<<endl;

cin>>choose;

}

//End catch invalid input choose

//Utilize switch to implement the menu

switch(choose) { //Begin switch choose

case 1:{

///////////////////////////////Component Calculator/////////////////////////////

cout<<"Program 1:New Component Longitudinal CG Calculator";

cout<<endl;

do {

//START CODE//////////////////////////////

//Prompt

cout<<setfill('-')<<setw(40)<<"-"<<endl;

cout<<" Welcome to Longitudinal CG Calculator!"<<endl;

cout<<setfill('-')<<setw(40)<<"-"<<endl;

cout<<setfill('-');

cout<<"This CG calculator is meant to aid you in recalculating \n";

cout<<"aircraft empty weight CG when you remove, add or replace \n";

cout<<"a permanent piece of equipment, as defined in the \n";

cout<<"Aircraft Equipment List."<<endl<<endl;

//Prompt for user to select ACFT to modify, or make their own

cout<<"Select which option you would prefer:"<<endl;

cout<<"1. Enter my own aircraft data"<<endl;

cout<<"2. Cessna 172M Profile"<<endl;

//Menu 1 Input Variable

unsigned short menu1;

//User input for menu1

cin>>menu1;

//Catch invalid input menu1

while (!(menu1>=1&&menu1<=2)) {

cin.clear();

cin.ignore();

cout<<"Not an option! Please re-enter: "<<endl;

cin>>menu1;

}

//Catch invalid input menu1

switch (menu1) { /////Menu1 Switch start

//Variables declared for CG calculations

//Planning to turn this into a function later

//Basic Aircraft Weight

float acftW;

//Aircraft moment comes from taking all aircraft longitudinal

//station numbers (in inches) and multiplying their weight by

//their ARM (distance from the DATUM)

float acftMom;

//CG is calculated by dividing aircraft moment by aircraft

//weight. The purpose here is to determine Basic Empty CG.

//So fuel, cargo and passengers are considered to be at 0.

float acftCG;

//Place holder for the weight of an item being removed

float itemOld;

//The location of the item being removed, in reference to

//manufacture specified DATUM point.

float armOld;

//Weight of new item

float itemNew;

//Location of new item, if modified. (EX. longer starter)

float armNew;

//Menu character

char armChoi;

case 1:{

//Retrieve Aircraft Weight

cout<<"Please enter aircraft empty weight, in lbs\n";

cout<<"to 1/10 of a lb accuracy. (xxxx.x)\n";

cout<<"NOTE: Info on acft Type Certificate Data Sheet\n";

cin>>acftW;

//Retrieve Aircraft Moment

cout<<"Please enter aircraft empty moment,\n";

cout<<"to 1/10 accuracy. (xxxx.x)\n";

cout<<"NOTE: Info on acft Type Certificate Data Sheet\n";

cin>>acftMom;

//Calculate CG

acftCG=(acftMom/acftW);

cout<<endl;

//Display CG

cout<<"Current CG is "<<acftCG<<"."<<endl;

//Ask what user wants to do to aircraft

cout<<"Are you adding, removing or replacing an item?\n";

cout<<"Type in option above in lowercase letters"<<endl;

cout<<"Type exit to leave or reset CG calculator."<<endl;

string opt1;

cin>>opt1;

do {

//Error catching loop opt1

while (opt1!="removing"&&opt1!="adding"&&opt1!=

"replacing"&&opt1!="exit"){

cin.clear();

cin.ignore();

cout<<"Invalid input, re-enter:"<<endl<<endl;

cout<<"Please type one of the following options: \n";

cout<<"removing \nadding \nreplacing\n";

cout<<"Type in option above in lowercase letters"<<endl;

cout<<"Or type exit to leave or reset CG calculator"<<endl;

cin>>opt1;

cout<<endl<<endl;

}

//End error catching loop opt1

//Removing option will only remove an item.

//COUT statements self explanatory

if (opt1=="removing") {

cout<<"You're removing an item"<<endl;

cout<<"Enter weight, in lbs to 1/10 lb accuracy\n";

cout<<"of item being removed"<<endl;

cin>>itemNew;

cout<<"Determining ARM: Is the item located\n";

cout<<"FWD or AFT of Acft Datum?"<<endl;

cout<<"F for FWD, A for AFT: ";

cin>>armChoi;

//Error catch statement for FWD & AWF

//Important that the user has to input this

//because it sets the ARM to positive or negative

do {

cout<<"Error! Invalid Input!"<<endl<<endl;

cout<<"Determining ARM: Is the item located\n";

cout<<"FWD or AFT of Acft Datum?"<<endl;

cout<<"F for FWD, A for AFT: ";

cin>>armChoi;

} while (armChoi!='f'&&armChoi!='F'

&&armChoi!='a'&&armChoi!='A');

//end error statement

cout<<"Enter ARM (distance from Datum in inches)\n";

cout<<"of item being removed, to 1/10th of an inch"<<endl;

cin>>armOld;

//set ARM to negative if item was FWD of datum

if (armChoi=='f'||armChoi=='F') armOld=(armOld\*-1);

//Calculations & Display

acftW=(acftW)-(itemNew)+.01;

cout<<setw(20)<<left<<"New Acft Weight: ";

cout<<setw(20)<<right<<acftW<<endl;

acftMom=acftMom-(itemNew\*armOld)+.01;

cout<<setw(20)<<left<<"New Acft Mom: ";

cout<<setw(20)<<right<<acftMom<<endl;

acftCG=(acftMom/acftW)+.01;

cout<<setw(20)<<left<<"New Acft CG is: ";

cout<<setw(20)<<right<<acftCG<<endl<<endl;

}

//Adding an item

if (opt1=="adding") {

cout<<"You're adding an item"<<endl;

cout<<"Enter weight, in lbs to 1/10 lb accuracy\n";

cout<<"of item being added"<<endl;

cin>>itemNew;

cout<<"Determining ARM: Is the item located\n";

cout<<"FWD or AFT of Acft Datum?"<<endl;

cout<<"F for FWD, A for AFT: ";

cin>>armChoi;

//Error catch statement ARM location

while (armChoi!='f'&&armChoi!='F'

&&armChoi!='a'&&armChoi!='A') {

cout<<"Error! Invalid Input!"<<endl<<endl;

cout<<"Determining ARM: Is the item located\n";

cout<<"FWD or AFT of Acft Datum?"<<endl;

cout<<"F for FWD, A for AFT: ";

cin>>armChoi;

}

//End error catch statement

cout<<"Enter ARM (distance from Datum in inches)\n";

cout<<"of item being removed, to 1/10th of an inch"<<endl;

cin>>armOld;

//Set ARM to negative if item FWD of datum

if (armChoi=='f'||armChoi=='F') armOld=(armOld\*-1);

//Calculations & display

acftW=(acftW)+(itemNew)+.01;

cout<<setw(20)<<left<<"New Acft Weight: ";

cout<<setw(20)<<right<<acftW<<endl;

acftMom=acftMom+(itemNew\*armOld)+.01;

cout<<setw(20)<<left<<"New Acft Mom: ";

cout<<setw(20)<<right<<acftMom<<endl;

acftCG=(acftMom/acftW)+.01;

cout<<setw(20)<<left<<"New Acft CG is: ";

cout<<setw(20)<<right<<acftCG<<endl<<endl;

}

//Replacing an item

if (opt1=="replacing") {

cout<<"You're replacing an item"<<endl;

cout<<"Determining ARM: Is the item located\n";

cout<<"FWD or AFT of Acft Datum?"<<endl;

cout<<"F for FWD, A for AFT: ";

cin>>armChoi;

//Error catching statement

do {

cout<<"Error! Invalid Input!"<<endl<<endl;

cout<<"Determining ARM: Is the item located\n";

cout<<"FWD or AFT of Acft Datum?"<<endl;

cout<<"F for FWD, A for AFT: ";

cin>>armChoi;

} while (armChoi!='f'&&armChoi!='F'

&&armChoi!='a'&&armChoi!='A');

//End error catching statement

cout<<"Enter ARM (distance from Datum in inches)\n";

cout<<"of item being removed, to 1/10th of an inch"<<endl;

cin>>armOld;

//Set ARM to negative if FWD of Datum

if (armChoi=='f'||armChoi=='F') armOld=(armOld\*-1);

cout<<"Was the ARM moved forward or aft?\n";

cout<<"Y for Yes, any other key for No:"<<endl;

cin>>armChoi;

armNew=armOld;

if (armChoi=='y'||armChoi=='Y') {

do {

cout<<"Forward, or Aft?"<<endl;

cout<<"F for FWD, A for AFT"<<endl;

cin>>armChoi;

if (armChoi=='f'||armChoi=='F') {

cout<<"Enter distance moved in inches\n";

cin>>armNew;

armOld=armOld-armNew;

}

else if (armChoi=='a'||armChoi=='A') {

cout<<"Enter distance moved in inches\n";

cin>>armNew;

armOld=armOld+armNew;

}

else {

cout<<"Error.";

}

} while (armChoi!='f'&&armChoi!='F'

&&armChoi!='a'&&armChoi!='A');

}

armOld=armNew;

cout<<"Enter the item's original weight"<<endl;

cout<<"Enter weight, in lbs to 1/10 lb accuracy\n";

cin>>itemOld;

acftMom=(acftMom)+(-itemOld\*armOld)+.01;

cout<<"Acft mom="<<acftMom<<endl;

cout<<"Enter weight of new item"<<endl;

cout<<"Enter weight, in lbs to 1/10 lb accuracy\n";

cin>>itemNew;

acftW=(acftW-itemOld)+(itemNew)+.01;

cout<<setw(20)<<left<<"New Acft Weight: ";

cout<<setw(20)<<right<<acftW<<endl;

acftMom=acftMom+(itemNew\*armOld)+.01;

cout<<setw(20)<<left<<"New Acft Mom: ";

cout<<setw(20)<<right<<acftMom<<endl;

acftCG=(acftMom/acftW)+.01;

cout<<setw(20)<<left<<"New Acft CG is: ";

cout<<setw(20)<<right<<acftCG<<endl<<endl;

}

cout<<"Are you adding, removing or replacing another item?"<<endl;

cout<<"Type in option above in lowercase letters"<<endl;

cout<<"Type exit to leave or reset CG calculator."<<endl;

cin>>opt1;

while (opt1!="removing"&&opt1!="adding"&&opt1!=

"replacing"&&opt1!="exit"){

cin.clear();

cin.ignore();

cout<<"Invalid input, re-enter:"<<endl<<endl;

cout<<"Please type one of the following options: \n";

cout<<"removing \nadding \nreplacing\n";

cout<<"Type in option above in lowercase letters"<<endl;

cout<<"Or type exit to leave or reset CG calculator"<<endl;

cin>>opt1;

cout<<endl<<endl; }

} while (opt1!="exit");

}

NOTE: Acft 172 Profile Left Out - Redundant Code; only difference is that it sets values into acftW & acftMom at the beginning, versus asking user

CG Calculator Test:

} ////////Menu 1 Switch End

//FINISH CODE/////////////////////////////

cout<<"\n\n";

cout<<"CG Calculator is reset."<<endl;

cout<<"Would you like to run calculator again?"<<endl;

cout<<"Type N for No and to return to main menu."<<endl;

cout<<"Press any other key to run calculator again."<<endl;

cin>>exit;

} while ((exit!='n')&&(exit!='N')); /////////do while loop

cout<<endl;

cout<<endl;

cout<<"End Program 1"<<endl;break;

} //End option 1

case 2:{

////////////////////////////////PROBLEM 2///////////////////////////////////////

cout<<"Welcome to Option 2.";

cout<<endl;

do {

//START CODE//////////////////////////////

//Randomly choose a sequence start

float acftW=0; //weight

float acftMom=0; //moment

float acftCG=0; //mom/weight

float testVal=0; //user input of new CG

unsigned short itemW=0; //weight of new item

unsigned short itemArm=0; //location of new item

srand(static\_cast<unsigned int>(time(0))); // time seed for rand

cout<<"Aircraft Weight: "<<endl; //display weight of 1000 to 1049

acftW=rand()%500+1000;

cout<<acftW<<" lbs"<<endl;

cout<<"Aircraft Moment: "<<endl; //display mom of 30000 to 30999

acftMom=rand()%1000+30000;

cout<<acftMom<<endl;

itemW=rand()%9+20; //display starting CG and create itemW 20-29

cout<<"Starting CG is: "<<endl;

cout<<acftMom/acftW<<endl<<endl;

cout<<"Item added weighs:"<<endl;

cout<<itemW<<" lbs"<<endl;

itemArm=rand()%9+1; //display item location 2-11

cout<<"Located "<<itemArm<<" inches aft of Datum"<<endl;

cout<<"Enter new aircraft CG: (to 1 decimal point)"<<endl;

//User inputs their calculation

cin>>testVal;

//Computer calculation for new CG

acftMom=acftMom+(itemW\*itemArm); //Add new items moment

acftW=acftW+(itemW); // add new items weight

acftCG=acftMom/acftW; //new cg

cout<<"Correct new CG is: "<<endl; //output to user

cout<<acftCG<<endl;

//FINISH CODE/////////////////////////////

cout<<"\n\n";

cout<<"Would you like to run again? N for No."<<endl;

cout<<"Press any other key to run again."<<endl;

cin>>exit;

} while ((exit!='n')&&(exit!='N'));

cout<<endl;

cout<<endl;

cout<<"End problem 2"<<endl;break;

} //End option 2

case 3: {

cout<<"Happy Landings!"<<endl;break;

}

} // switch statement end bracket

} while (choose!=3);

//Exit Stage Right

return 0;

}