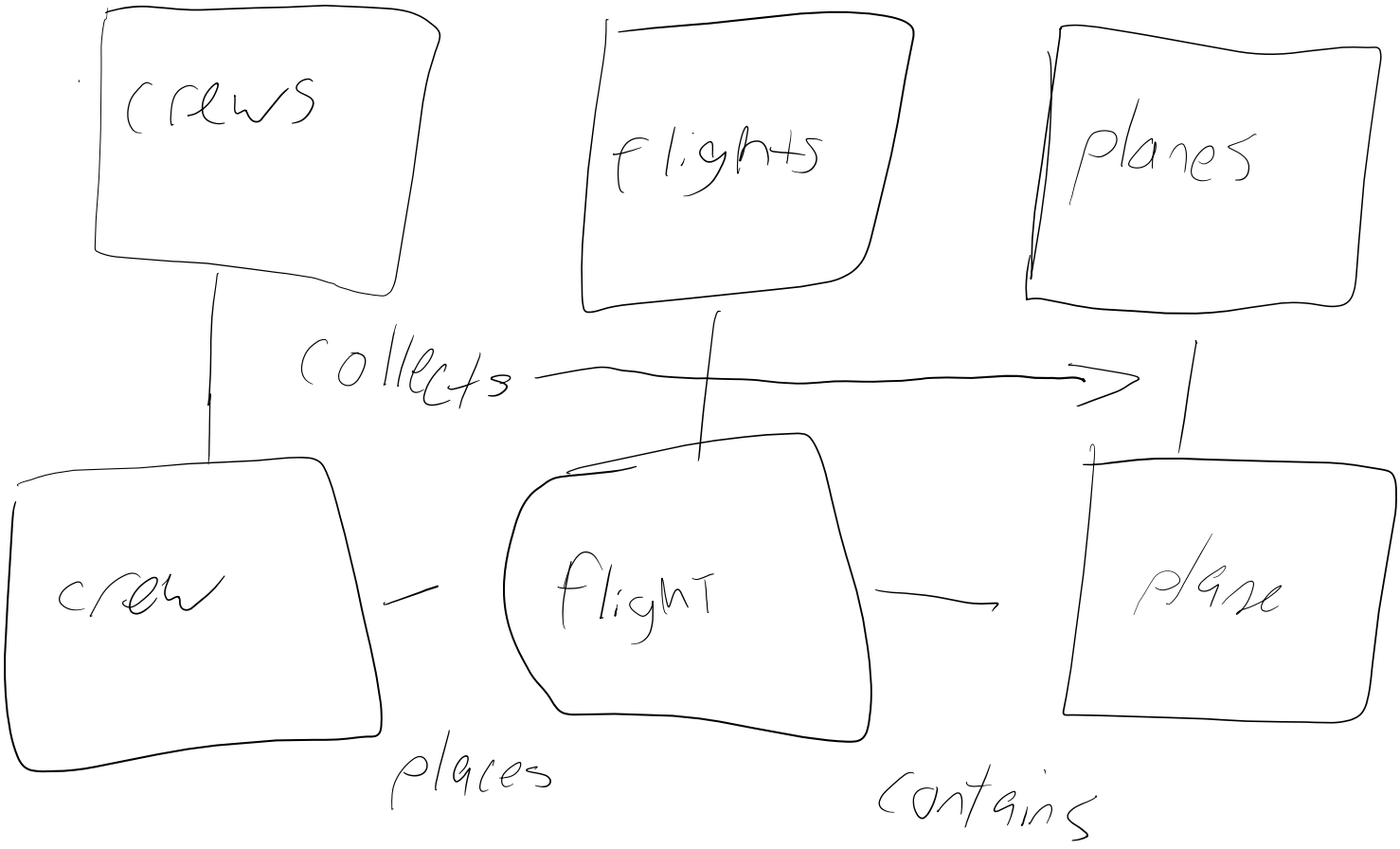
Homework 3 algorithm



class contents

Plane

- String make
- String model
- String tail number
- Int number of seats
- Int range
- String status

Crew

- String name
- Int id
- String type
- String status

Flight

- String tail number
- Int pilot id
- Int copilot id
- Int crew ids
- String start date and time
- String end date and time
- String starting airport code
- String ending airport code
- Int number of passengers
- String status

Planes

- Int count
- Vector of planes
- Add plane
- Edit plane
- Delete plane
- Find plane
- print planes

Crews

- int count
- vector of crews
- add crew
- edit crew
- delete crew
- find crew
- print crews

Flights

- int count
- vector of flights
- add flight
- edit flight
- delete flight
- find flight
- print flight

pseudo code

class plane crew and flight

- standard mutators and accessors

planes

- vector <plane> planelist
- add plane
  - prompt user to fill out the required info
  - user will enter
    - make
    - model
    - tail number
    - number of seats
    - range
    - number of seats
- delete plane
  - user will input tail number of plain
  - pass tail number to search function
  - once found use erase function of vector to remove element
- edit plane

    - user will be prompted to enter the tail number of the plane they want to edit
    - user then will then be asked what they want to change about the plane
    - user will then select off the menu what they want to change'
    - once user selected what they wanted to change he will enter the new info
    - the new info will be put into the vector replacing what is already there
    - display new info for the user
- find plane
    - user will enter the tail number
    - use search function in <algorithm> to find plane
- print planes
    - use for loop to display all the info for each plane using <iomanip> for formatting

crews class add delete edit find and print will be the same in terms of logic

The Flights will also be mostly the same

- when adding the necessary information there needs to be a function that checks if the crew or plane exists and is available
    - the plane must be available meaning it not already scheduled for another flight same goes for the crew members
    - this will be done with a bool function that return true if the plane/crew is available at the selected time
    - to do this we need to include the other class files and create the objects in this file to call the necessary function to check availability of the elements needed for the flights class

Changes made in Homework 4

- Instead of using vectors for the collection I instead used maps.
    - For Flights the Flight ID was used as the key
    - For Crews the Crew member ID was used as the key
    - For Planes the Tail Number was used as the key.
- The menu is divided into sub menus:
    - ADD
    - EDIT
    - DELETE
    - PRINT ENTITY IN COLLECTION
    - PRINT ENTIRE COLLECTION

- Each menu then displays a sub menu:
  - Planes
  - Crews
  - Flights
- The menu loops using recursive functions instead using a while loop
  - Each case in the switch statement calls the function for that specific menu again after modifying/displaying the collection

```
switch (i)
	{
		case 0:PL.store(); CW.store(); FL.store(); EXIT_SUCCESS;	break;
		case 1: cout << "ADD SELECTED\n" << endl; addMenu(); break;
		case 2: cout << "EDIT SELECTED\n" << endl; editMenu(); break;
		case 3:	cout << "DELETE SELECTED\n" << endl; deleteMenu(); break;
		case 4: cout << "PRINT ALL SELECTED\n" << endl; printALL();	break;
		case 5:	cout << "PRINT SINGLE SELECTED\n" << endl; printSingle();
			break;
		default: cout << "INVALID CHOICE, ";	mainMenu();	break;
		}
```

This is what the main menu looks like

- The option that the user chooses takes them to that desired submenu
- The user can exit the submenu by entering 0 taking them back to the main menu

- Storing and loading
  - The Store and Load functions are a member function in each collection.
    - Each collections Load function runs at the start of the programs
    - Each collection has its own data file that it locates and read from
      - If not found it will display it was not found
    - Each function also contains a store function that run before termination of the program
- Planes and Crews has a member function called AddToFlights
  - The member function is used to ensure the user enter valid info contained in the other collections
  - These functions have a call by reference parameters that modify the value of the variable passed into it
  - These modified variables are then passed into the ADD function for Flights to add to its collection.