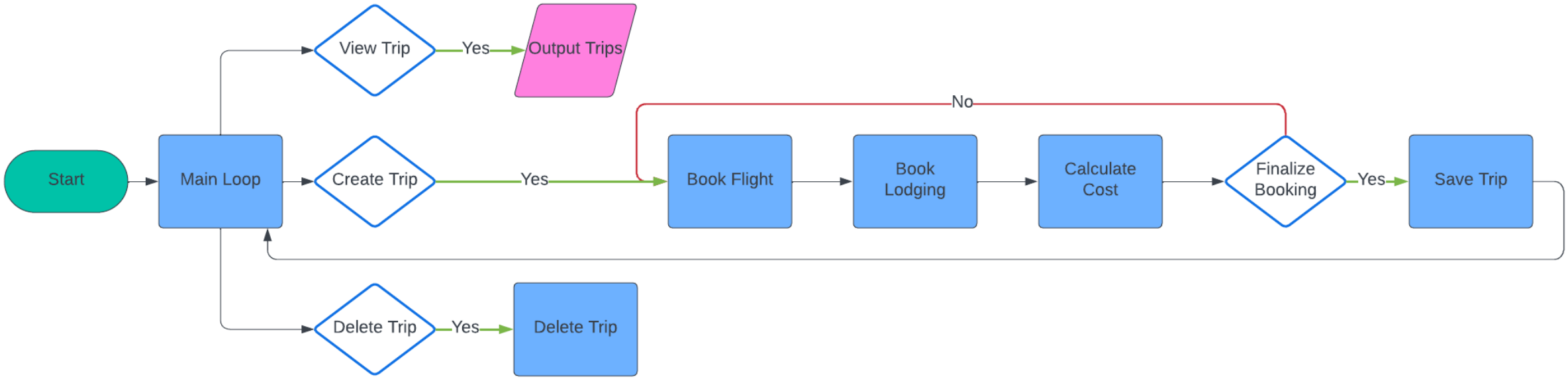
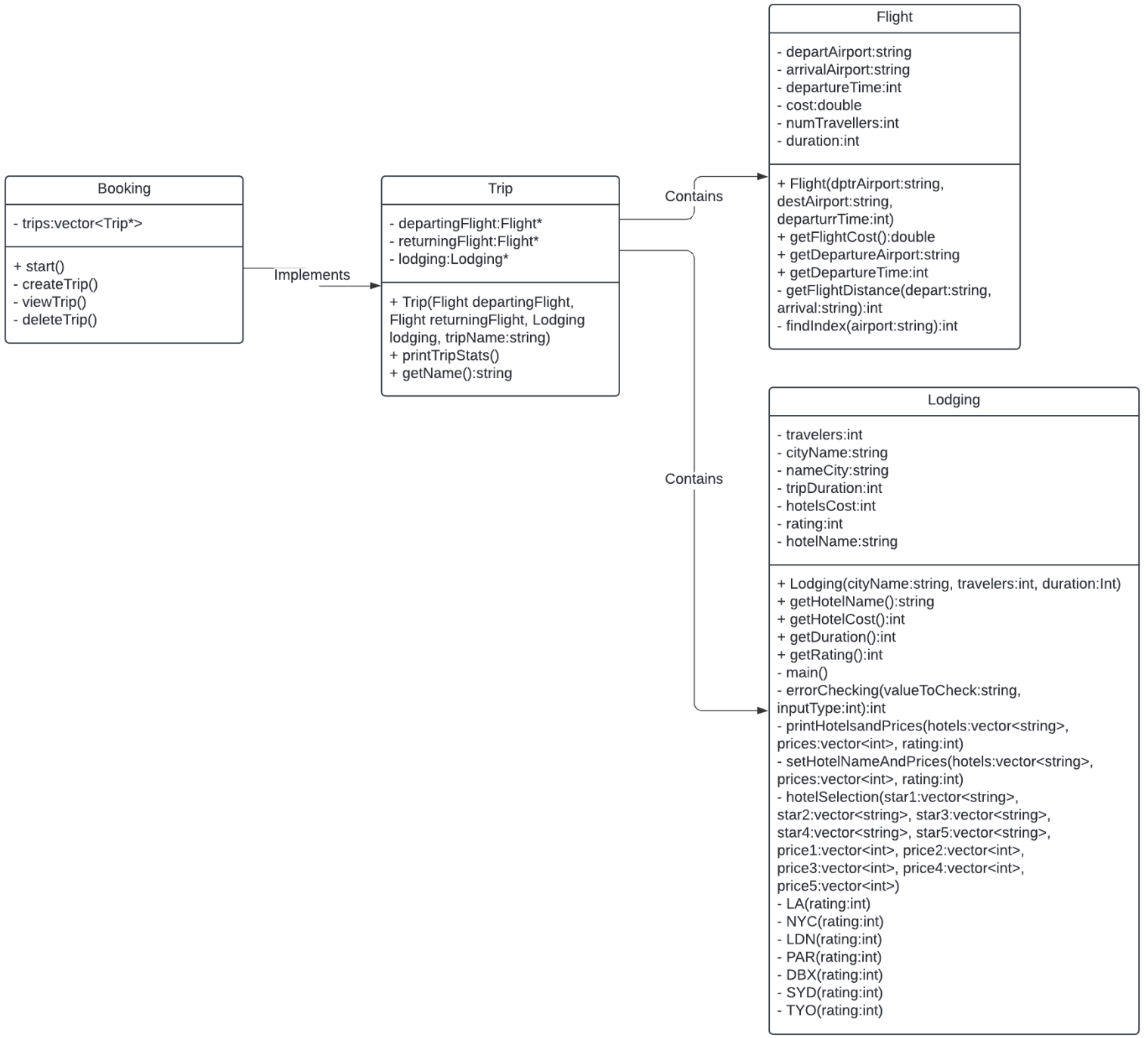
Holiday Booking System

Jaime Garcia, Thomas Mestrov, Nathan Neidigh, Joseph Pennock

Flow Chart


Start
Main Loop
View Trip
Yes
Output Trips
Create Trip
Yes
No
Book Flight
Book Lodging
Calculate Cost
Finalize Booking
Yes
Save Trip
Delete Trip
Yes
Delete Trip



UML Class Diagrams

```
Variables= 25
Cubes= 10000000
terminate called after throwing an instance of 'std::bad_alloc'
  what():  std::bad_alloc

This application has requested the Runtime to terminate it in an unusual way.
Please contact the application's support team for more information.

Process returned 3 (0x3)   execution time : 14.655 s
Press any key to continue.
```

Complications

```
    if (cityName == "dubai") {
        DBX(rating: fixedHotelRating);
    }

    if (cityName == "sydney") {
        SYD(rating: fixedHotelRating);
    }

    if (cityName == "tokyo") {
        TYO(rating: fixedHotelRating);
    }
}


void Lodging::LA(int rating) {
    // Using paralell arrays to store hotel names and hotel prices per each rating.
    std::vector<std::string> star1 = {[0]="Regal Inn", [1]="Economy Inn", [2]="Willow Tree Inn", [3]="Sea Breeze Inn", [4]="Banana Bungalow West Hollywood"};
    std::vector<std::string> star2 = {[0]="Antonio Hotel", [1]="Los Angeles Inn & Suites", [2]="City Center Hotel", [3]="Value Inn Hollywood", [4]="Valley Inn"};
    std::vector<std::string> star3 = {[0]="LA Adventure Hotel", [1]="All Star Inn", [2]="Shelter Hotels", [3]="Palihotel Melrose Avenue", [4]="Sunrise Hotel San Pedro"};
    std::vector<std::string> star4 = {[0]="The Biltmore Los Angeles", [1]="Dream Hollywood", [2]="SIXTY Beverly Hills", [3]="Sofitel Los Angeles at Beverly Hills", [4]="Hilton Los Angeles Culver City"};
    std::vector<std::string> star5 = {[0]="Four Seasons Hotel", [1]="1 Hotel West Hollywood", [2]="The Beverly Hills Hotel", [3]="Shutters on the Beach", [4]="Beverly Wilshire"};
    std::vector<int> price1 = {[0]=89, [1]=120, [2]=97, [3]=73, [4]=60};
    std::vector<int> price2 = {[0]=58, [1]=89, [2]=77, [3]=95, [4]=90};
    std::vector<int> price3 = {[0]=85, [1]=116, [2]=86, [3]=245, [4]=111};
    std::vector<int> price4 = {[0]=160, [1]=212, [2]=285, [3]=283, [4]=205};
    std::vector<int> price5 = {[0]=926, [1]=427, [2]=1195, [3]=632, [4]=778};

    hotelSelection(star1, star2, star3, star4, star5, price1, price2, price3, price4, price5, rating);
}


void Lodging::NYC(int rating) {
    // Using paralell arrays to store hotel names and hotel prices per each rating.
    std::vector<std::string> star1 = {[0]="Bowery Grand Hotel", [1]="Wood Spring Suites Linden", [2]="West Side YMCA", [3]="U.S Pacific Hotel", [4]="Chelsea International Hotel"};
    std::vector<std::string> star2 = {[0]="Queens Hotel", [1]="Hotel St. James", [2]="Sheridan Hotel", [3]="Carlton Arms Hotel", [4]="Red Carpet Inn Brooklyn"};
    std::vector<std::string> star3 = {[0]="The Manhattan at Times Square", [1]="45 Times Square Hotel", [2]="Pod Times Square", [3]="The Gregorian Hotel", [4]="YOTEL NY Times Square"};
    std::vector<std::string> star4 = {[0]="Millennium Hilton", [1]="Warwick NY", [2]="Nilton NY Times Square", [3]="Hyatt Grand Central NY", [4]="The Empire Hotel"};
    std::vector<std::string> star5 = {[0]="The Times Square EDITION", [1]="The Laghman", [2]="The Knickerbocker", [3]="The St. Regis NY", [4]="The Bowery Hotel"};
    std::vector<int> price1 = {[0]=69, [1]=97, [2]=103, [3]=119, [4]=117};
    std::vector<int> price2 = {[0]=108, [1]=109, [2]=134, [3]=87, [4]=119};
    std::vector<int> price3 = {[0]=84, [1]=133, [2]=115, [3]=137, [4]=96};
    std::vector<int> price4 = {[0]=152, [1]=133, [2]=150, [3]=161, [4]=136};
    std::vector<int> price5 = {[0]=305, [1]=672, [2]=224, [3]=820, [4]=564};

    hotelSelection(star1, star2, star3, star4, star5, price1, price2, price3, price4, price5, rating);
}


void Lodging::LDN(int rating) {
    // Using paralell arrays to store hotel names and hotel prices per each rating.
    std::vector<std::string> star1 = {[0]="Corbigoe Hotel", [1]="Aquarius Hotel", [2]="St. George Hotel", [3]="Holly House Hotel LDN", [4]="The London Hotel"};
    std::vector<std::string> star2 = {[0]="Paramount Hotel", [1]="Star Hotel", [2]="Sheriff Hotel", [3]="European Hotel", [4]="Hyde Park Court Hotel"};
    std::vector<std::string> star3 = {[0]="Sapphire Hotel LDN", [1]="Rushmore Hotel", [2]="Travelodge LDN", [3]="King Solomon Hotel", [4]="Zedwell Piccadilly Circus"};
    std::vector<std::string> star4 = {[0]="Britannia International Hotel", [1]="Hilton LDN Hyde Park", [2]="The Crown LDN Hotel", [3]="The Tower Hotel", [4]="Merit Kensington Hotel"};
    std::vector<std::string> star5 = {[0]="Bulgari Hotel", [1]="The Ritz LDN", [2]="Shangri-La The Shard", [3]="Claridge's", [4]="Rosewood LDN"};
    std::vector<int> price1 = {[0]=35, [1]=47, [2]=46, [3]=57, [4]=60};
    std::vector<int> price2 = {[0]=48, [1]=64, [2]=50, [3]=63, [4]=53};
    std::vector<int> price3 = {[0]=51, [1]=61, [2]=62, [3]=52, [4]=60};
    std::vector<int> price4 = {[0]=57, [1]=158, [2]=82, [3]=148, [4]=100};
    std::vector<int> price5 = {[0]=1114, [1]=880, [2]=663, [3]=964, [4]=614};
```

```
    isInteger1 = true;
    for (char c : input1) {
      if (!isdigit(c)) {
        isInteger1 = false;
        break;
      }
    }

    if (!isInteger1) {
      std::cout << "Invalid input. Please enter a valid integer." << std::endl;
    }
  } while (!isInteger1);

  travelers = std::stoi(str: input1);

  std::string input2;
  bool isInteger2;

  do {
    std::cout << "Enter duration of stay (number of days): ";
    std::getline(&is: std::cin, &str: input2);

    isInteger2 = true;
    for (char c : input2) {
      if (!isdigit(c)) {
        isInteger2 = false;
        break;
      }
    }

    if (!isInteger2) {
      std::cout << "Invalid input. Please enter a valid integer." << std::endl;
    }
  } while (!isInteger2);

  duration = std::stoi(str: input2);

  std::string tripName;

  std::cout << std::endl;
  std::cout << "Enter name for the trip: ";
  std::getline(&is: std::cin, &str: tripName);
  std::cout << std::endl;
  std::cout << "travlers: " << travelers << std::endl;

  Flight *departingFlight = new
      Flight(departAirport: departingCity, arrivalAirport: destinationCity, departureTime: departTime, numTravelers: travelers);
  Flight *returningFlight = new
      Flight(departAirport: destinationCity, arrivalAirport: departingCity, departureTime: departTime, numTravelers: travelers);
  Lodging *lodge = new Lodging(cityName: destinationCity, travelers, duration);

  Trip *t = new Trip(departingFlight, returningFlight, lodging: lodge, tripName);
  this->departingCity = departingCity;
  this->destinationCity = destinationCity;
  this->departTime = departTime;
  this->travelers = travelers;
  this->duration = duration;

  this->trips.push_back(x: t);
}
```

Questions?