Final Spec Sheet – Caleb Flegel

**Use Cases**

|  |  |  |
| --- | --- | --- |
| Step | User Action | Program Response |
| 1 | User starts program |  |
| 2 |  | Program asks for a visit rate |
| 3 | User enters number between 1 and 60 |  |
| 4 |  | Program accepts answer stores the infection rate |
| 5 |  | Program prompts user for the number of doctors |
| 6 | User enters number more than 0 |  |
| 7 |  | Program accepts answer and saves the requested number |
| 8 |  | Program prompts user for the requested number of nurses |
| 9 | User enters a number more than |  |
| 10 |  | Program accepts response for the number of nurses |
| 11 |  | Program starts the weekly simulation using the given variables |
| 12 |  | Program completes simulation |
| 13 |  | Program asks what the user wants to do with the records |
| 14 | User enters “see all records” |  |
| 15 |  | Program will print out all hospital records to the console |
| 16 | User enters “name lookup” |  |
| 17 |  | Program will prompt the user for a last name |
| 18 | User enters a name that exists the in the hospital records |  |
| 19 |  | Program prints all records with the inputted name |
| 20 | User enters a last name not in the record |  |
| 21 |  | Program tells user the last name wasn’t found |
| 22 | User enters “end” |  |
| 23 |  | The program is shut down |
| 24 | User enters an invalid input |  |
| 25 |  | Program tells the user their input doesn’t exist |

**Psuedocode**

FUNCTION Town::Town (int docNum, int nurNum, int sickChance)

Make class variable emergencyChnce equal sickChance

Create a new hospital for class variable mainHospital

Call mainHospital’s class method addProviders with docNum and nurNum

Call the pullCitizens method to fill the residentList Map

ENDFUNCTION

FUNCTION Town::~Town()

Delete the mainHospital object point created in the constructor

END FUNCTION

FUNCTION void Town::pullCitizens()

Create strings first and last

Create firstInput and lastInput ifstreams

Open the first and last name files with firstInput and lastInput respectively

WHILE it is not the end of lastInput

Collect the next string from lastInput and assign it to last

Collect the next string from firstInput and assign it to first

Add the new resident name to the residentList map using first and last

ENDWHILE

ENDFUNCTION

FUNCTION Hospital Town::getHospital()

Return mainHospital

ENDFUNCTION

FUNCTION static Time Town::getTIme()

Return currentTime

ENDFUNCTION

FUNCTION static string Town::getFirst(string lastName)

Return the name from the resident list map [lastName]

ENDFUNCTION

FUNCTION void Town::sickCheck()

Create an iterator to go through the residentList map and start it at the beginning

FOR i -> 0 to residentList’s size

IF a random number mod (120,000 / emergencyChnce) is 0 (Random chance to get sick)

IF the citizen if not a patient in the patientQueue by calling patCheck using the last name

Add the resident to the patient queue using newPatient with the last name

ENDIF

ENDIF

Move the iterator to the next spot in the map

ENDFOR

ENDFUNCTION

FUNCTION void Town::runWeek()

WHILE currentTIme’s day is less than 7

Call sickCheck to determine which citizens get sick

Call mainHospital’s providerDuties to have the provider treat patients

Call currentTime’s advanceTime to advance the time

IF currentTime’s minute is 0

Print the current time

ENDIF

ENDFUNCTION

FUNCTION Hospital::HOSPITAL()

ENDFUNCTION

FUNCTION Hospital::~Hospital()

Delete providerList

ENDFUNCTION

FUNCTION Hospital::toUppercase (string& input)

FOR each letter of input

IF the letter is lower case

Subtract 32 from the letter to convert to uppercase

ENDIF

ENDFOR

ENDFUNCTION

FUNCTION vector<Record> Hospital::getRectords()

Return patientRecords

ENDFUNCTION

FUNCTION int Hospital::getPatientCount()

Return listOfPatients’s size

ENDFUCNTION

FUNCTION Patient Hospital::getPatient

Initialize temp and assign it to the top patient off of the listOfPatients queue

Pop the top patient off of the queue

Return temp

ENDFUNCTION

FUNCTION void Hospital::pushPatient(Patient P)

Add P to the listOfPatient’s queue

ENDFUNCTION

FUNCTION void Hospital::addProviders(int doctors, int nurses)

FOR i -> 0 to doctors

Add a new doctor to providerList

ENDFOR

FOR I -> 0 to nurses

Add a new nurse to providerList

ENDFOR

ENDFUNCTION

FUNCTION void Hospital::newPatient (string lastName)

Push a new Patient with the name of lastName to the listOfPatients

ENDFUNCTION

FUNCTION bool Hospital::searchPatient (string lastName)

Call to uppercase to convert lastName to all uppercase letters

Create a fullName string by combining lastName and the firstName (found by calling Town::getFirst)

FOR i -> 0 to patientRecords’s size

IF fname is patienRecords.getName() at i

Return true

ENDIF

ENDFOR

Return false

ENDFUNCTION

FUNCTION void Hospital::printNameRecord (string lastName)

Call toUppercase with lastName to convert lastName to all uppercase letters

Get the full name fName by getting the first name of the resident by calling lastName

FOR i -> 0 to patientRecords’ size

IF the name of the person in the record at I is fName

Print the record

ENDIF

ENDFOR

ENDFUNCTION

FUNCTION static void Hospital::addTreatment (Patient patient)

Push a record with the patient to the back of patientRecords

ENDFUNCTION

FUNCTION Time Hospital::getAvgTreatment()

Create total and temp Time objects

FOR i -> 0 to size of patientRecords

Assign temp to the visit time of the record at i

Add temp’s time to the total time

ENDFOR

Convert total to minutes and store the minute number in totMinutes

Divide totMinutes by the size of patientRecords

Assign temp to a new Time object with the average number of minutes

Adjust temp to convert minutes to hour and days if needed

Return temp

ENDFUNCTION

FUNCTION bool Hospital::patCheck(string name)

Create a check bool flag, starting at false

Create a vector of Patients called reset

FOR i->0 to size of listOfPatients

Create a temp Patient from the top of the patient list

Add temp to the end of reset

Pop the top of the patient list

IF the name of Patient temp is name

Make check true

Break the for loop

ENDIF

ENDFOR

FOR i->0 to size of reset

Push the reset Patient at i to the patient list

ENDFOR

ENDFUNCTION

FUNCTION void Hospital::providerDuties()

FOR i->0 to size of providerList

IF the provider at providerList i is available

Call the setTreatment method for the provider

ELSE

Call the treatPatient method for the provider

ENDIF

ENDFUNCTION

FUNCTION TIME::Time (int day, int hour, int minute

Assign the day, hour, and minute variables to the arguments

ENDFUNCTION

FUNCTION int TIME::getDay()

Return the day class variable

ENDFUNCTION

FUNCTION int TIME::getHour()

Return the hour class variable

ENDFUNCTION

FUNCTION int TIME::getMinute()

Return the minute class variable

ENDFUNCTION

FUNCTION void TIME::operator() ()

WHILE minute is larger than 60

Add one to hour

Subtract 60 from minutes

ENDWHILE

WHILE hour is larger than 24

Add one to day

Subtract 24 from hour

ENDWHILE

ENDFUNCTION

FUNCTION int TIME::toMinutes(Time inpTime)

WHILE (inpTime’s day is larger than 0)

Add 24 to inpTime’s hour

Subtract one from inpTime’s day

ENDWHILE

WHILE inpTime’s hour is greater than 0

Add 60 to inpTime’s minute

Subtract one from inpTime’s hour

ENDWHILE

Return inpTime’s minute

ENDFUNCTION

FUNCTION void TIME::advanceTime()

Add one to minute

Readjust this time using () functor

ENDFUNCTION

FUNCTION Time TIME::operator + (Time& time1, Time& time2)

Create a newDay int by combining the day variables of time 1 and time 2

Create a newHour int by combining the hour variables of time 1 and time 2

Create a newMinute int by combining the minute variables of time 1 and time 2

Create a temp Time object using newDay, newHour, and newMinute

Readjust temp using ()

Return temp

ENDFUNCTION

FUNCTION Time TIME::operator+= (Time& time1, Time& time2)

Return time1 + time2

ENDFUNCTION

FUNCTION Time TIME::operator-(Time& time1, Time& time2)

Create a newDay int by subtracting time2’s day and time1’s day

Create a newHour int by subtracting time2’s hour and time1’s hour

Create a newMinute int by subtracting time2’s minute and time1’s minute

Create a temp Time object using newDay, newHour, and newMinute

Readjust temp using ()

Return temp

ENDFUNCTION

FUNCTION Time TIME::operator-= (Time& time1, Time& time2)

Return time1-time2

ENDFUNCTION

FUNCTION PATIENT::PATIENT ()

Set surname as “undefined”

Set priority to -1

ENDFUNCTION

FUNCTION PATIENT::Patient (string name)

Set startTime to the current time using Town::getTime()

Generate a random number between 0 and 9 and store it in priorityChk

SWITCH (priorityChk)

Case 0 thru 6

Set priority to a random number between 1 and 10

Break switch

Case 7 and 8

Set priority to random number between 11 and 15

Break switch

Case 9

Set priority to random number between 16 and 20

Break switch

Default

Break

ENDSWITCH

ENDFUNCTION

FUNCTION string PATIENT:: getName()

Return surname

ENDFUNCTION

FUNCTION int PATIENT:: getPriority()

Return priority

ENDFUNCTION

FUNCTION Time PATIENT:: getStartTime()

Return startTime

ENDFUNCTION

FUNCTION bool operator> (const Patent& pat1, const Patient& pat2)

Return if pat1’s priority is greater than pat2’s priority

ENDFUNCTION

FUNCTION bool operator< (const Patient& pat1, const Patient& pat2)

Return if pat1’s priority is less than pat2’s priority

ENDFUNCTION

FUNCTION RECORD::Record(Patient treatedPat)

Set name to treatedPat’s name plus the resident’s first name (gotten by using Town::getFirst())

Set priority to treatedPat’s priority

Create time temp1 and make it equal the current time

Create time temp2 and make it equal treatedPat’s start time

Set totalTime to temp1 – temp2

ENDFUNCTION

FUNCTION string RECORD::getName()

Return name

ENDFUNCTION

FUNCTION int RECORD::getPriority()

Return priority

ENDFUNCTION

FUNCTION Time RECORD::getTotalTime

Return totalTime

ENDFUNCTION

FUNCTION ostream& operator<< (ostream& out, Record& rec)

Print patient name and priority from rec

Print patient’s total visit time from rec

ENDFUNCTION

FUNCTION PROVIDER::Provider()

Set available to true and remainingTreatTime to -1

ENDFUNCTION

FUNCTION virtual void PROVIDER::setTreatment() = 0

ENDFUNCTION

FUNCTION void PROVIDER::treatPatient()

Subtract one from remainingTreatTime

IF (remainingTreatTime is less than 0)

Set available is true

Add current patient to Hospital’s patientRecord vector

ENDIF

ENDFUNCTION

FUNCTION bool PROVIDER::isAvailable()

Return available

ENDFUNCTION

FUNCTION Patient PROVIDER::getPatient()

Return currentPatient

ENDFUNCTION

FUNCTION Doctor::Doctor()

ENDFUNCTION

FUNCTION Doctor::setTreatment()

IF Hospital’s patientList queue has a size of 1 or more

Set currentPatient to the top patient of the patientList queue

Set remainingTreatTime to a random number between 1 and 20

Set Available to false

ENDIF

ENDFUNCTION

FUNCTION Nurse::Nurse()

ENDFUNCTION

FUNCTION Nurse::Nurse()

Create a temp patient

Create a vector of patients called passedPats

WHILE (Hospital’s patientList size is greater than 0)

Set temp to the top patient of the hospital patientList

IF (temp’s priority is less than 11)

Set the currentPatient to temp

Set remainingTreatTime to a random number between 1 and 10

Set available to false

Break the while loop

ELSE

Push temp to passedPats

ENDIF

ENDWHILE

FOR i->0 to passedPats’s size

Push patient at passedPats[i] to the hospital’s patient list

ENDFUNCTION