//Enhan Zhao 11097118 enz889 cmpt270

Testing:
The testing for all 3 classes are hard coded in the main function of each of the class file.
Testing for Patient, Ward, Surgeon, Doctor class can be done by running the main function of each class.
Testing for Patient and Doctor requires BasicDoctor and person class. Testing for Surgeon class requires DOctor class. Since these classes inherit from the super classes.
No test drivers were provided, so each main function needs to be called seperately.
If errors were found, a print statement will show in which method are errors.

Assignment status:

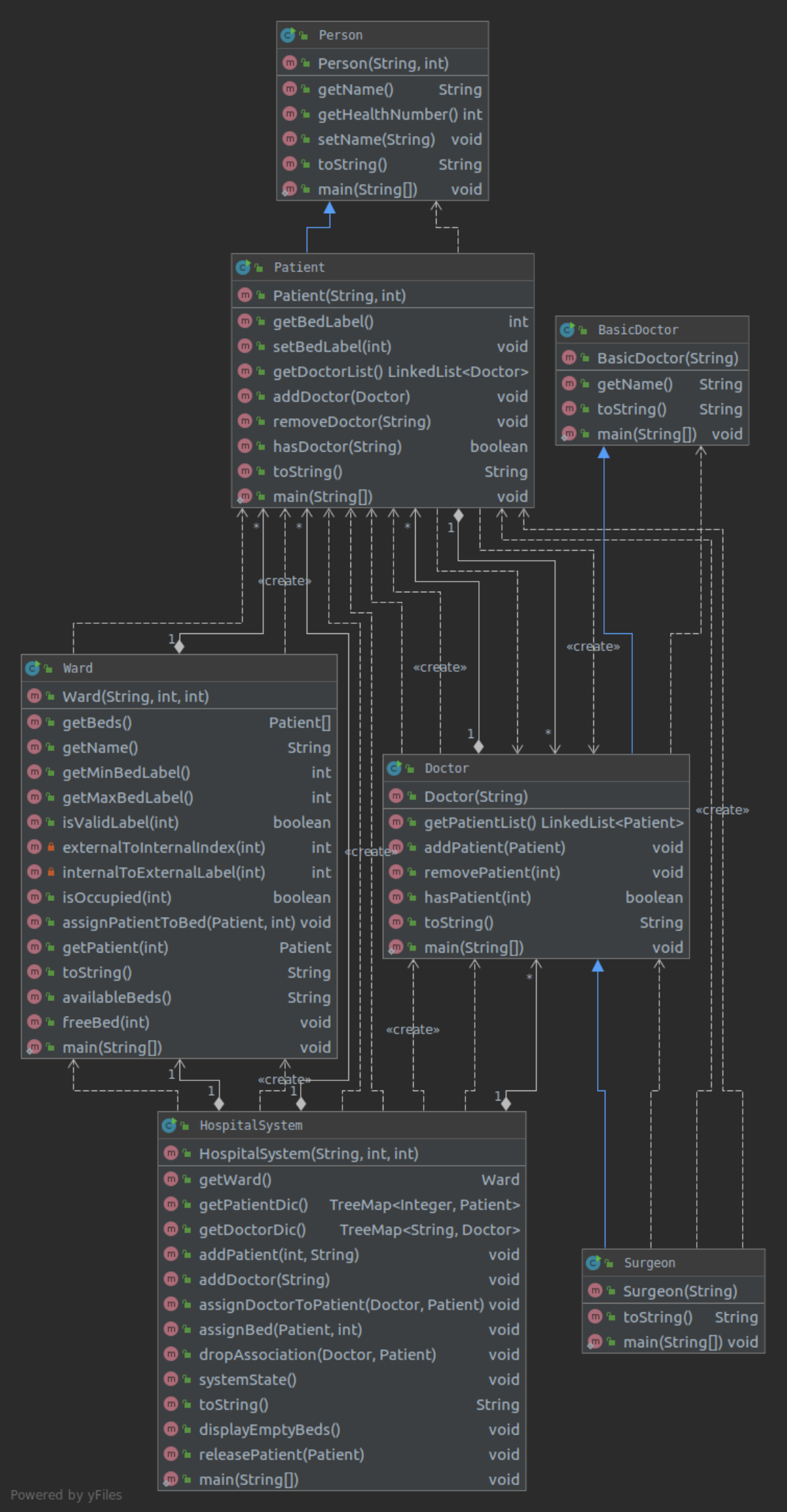
Every methods in each class works as intended, and the HospitalSystem functions well. However the system is not very robust. If more time is given, then every single method that
has an argument would have assert statements to check if the arguments are valid. This will make the program much more robust, but
this will also take a massive amount of time to complete.

Demonstration of Hospital System:////////////////////////////////

```
Enter hospital name:
Sask
Enter minimum bed label:
10
Enter maximum bed label:
20
1. quit
2. add a new patient to the system
3. add a new doctor to the system
4. assign a doctor to a patient
5. display the empty beds of the ward
6. assign a patient a bed
7. release a patient
8. drop doctor-patient association
9. display current system state
2
Adding patient. Please enter patient health card number
111
Adding patient. Please enter patient name:
Paul
1. quit
2. add a new patient to the system
3. add a new doctor to the system
4. assign a doctor to a patient
5. display the empty beds of the ward
6. assign a patient a bed
7. release a patient
8. drop doctor-patient association
9. display current system state
3
Adding doctor. Please enter doctor name:
Bob
1. quit
2. add a new patient to the system
3. add a new doctor to the system
4. assign a doctor to a patient
5. display the empty beds of the ward
6. assign a patient a bed
7. release a patient
```

```
8. drop doctor-patient association
9. display current system state
4
Assigning patient to doctor. Enter patient health card number:
111
Assigning patient to doctor. Enter doctor name:
Bob
1. quit
2. add a new patient to the system
3. add a new doctor to the system
4. assign a doctor to a patient
5. display the empty beds of the ward
6. assign a patient a bed
7. release a patient
8. drop doctor-patient association
9. display current system state
5

Ward Sask with capacity 11 has these empty beds:
bed10
bed11
bed12
bed13
bed14
bed15
bed16
bed17
bed18
bed19
bed20


1. quit
2. add a new patient to the system
3. add a new doctor to the system
4. assign a doctor to a patient
5. display the empty beds of the ward
6. assign a patient a bed
7. release a patient
8. drop doctor-patient association
9. display current system state
6
Assigning patient to bed. Enter patient health card number:
111
Assigning patient to bed. Enter empty bed label:
15
1. quit
2. add a new patient to the system
3. add a new doctor to the system
4. assign a doctor to a patient
5. display the empty beds of the ward
6. assign a patient a bed
7. release a patient
8. drop doctor-patient association
9. display current system state
8
Dropping patient-doctor association. Enter patient health card number:
111
Dropping patient-doctor association. Enter doctor name:
Bob
1. quit
2. add a new patient to the system
3. add a new doctor to the system
4. assign a doctor to a patient
```

```
5. display the empty beds of the ward
6. assign a patient a bed
7. release a patient
8. drop doctor-patient association
9. display current system state
7
Releasing Patient. Enter patient health card number:
111
1. quit
2. add a new patient to the system
3. add a new doctor to the system
4. assign a doctor to a patient
5. display the empty beds of the ward
6. assign a patient a bed
7. release a patient
8. drop doctor-patient association
9. display current system state
9
Doctor names:
Bob
Patient names:

Ward Sask with capacity 11 has the following patients:
bed 10:
bed 11:
bed 12:
bed 13:
bed 14:
bed 15:
bed 16:
bed 17:
bed 18:
bed 19:
bed 20:

1. quit
2. add a new patient to the system
3. add a new doctor to the system
4. assign a doctor to a patient
5. display the empty beds of the ward
6. assign a patient a bed
7. release a patient
8. drop doctor-patient association
9. display current system state
1
Logging off
```


Person
Person(String, int)
getName() String
getHealthNumber() int
setName(String) void
toString() String
main(String[]) void
Patient
Patient(String, int)
getBedLabel() int
setBedLabel(int) void
getDoctorList() LinkedList<Doctor>
addDoctor(Doctor) void
removeDoctor(String) void
hasDoctor(String) boolean
toString() String
main(String[]) void
BasicDoctor
BasicDoctor(String)
getName() String
toString() String
main(String[]) void
Ward
Ward(String, int, int)
getBeds() Patient[]
getName() String
getMinBedLabel() int
getMaxBedLabel() int
isValidLabel(int) boolean
externalToInternalIndex(int) int
internalToExternalLabel(int) int
isOccupied(int) boolean
assignPatientToBed(Patient, int) void
getPatient(int) Patient
toString() String
availableBeds() String
freeBed(int) void
main(String[]) void
Doctor
Doctor(String)
getPatientList() LinkedList<Patient>
addPatient(Patient) void
removePatient(int) void
hasPatient(int) boolean
toString() String
main(String[]) void
HospitalSystem
HospitalSystem(String, int, int)
getWard() Ward
getPatientDic() TreeMap<Integer, Patient>
getDoctorDic() TreeMap<String, Doctor>
addPatient(int, String) void
addDoctor(String) void
assignDoctorToPatient(Doctor, Patient) void
assignBed(Patient, int) void
dropAssociation(Doctor, Patient) void
systemState() void
toString() String
displayEmptyBeds() void
releasePatient(Patient) void
main(String[]) void
Surgeon
Surgeon(String)
toString() String
main(String[]) void
«create»
1
*
Powered by yFiles