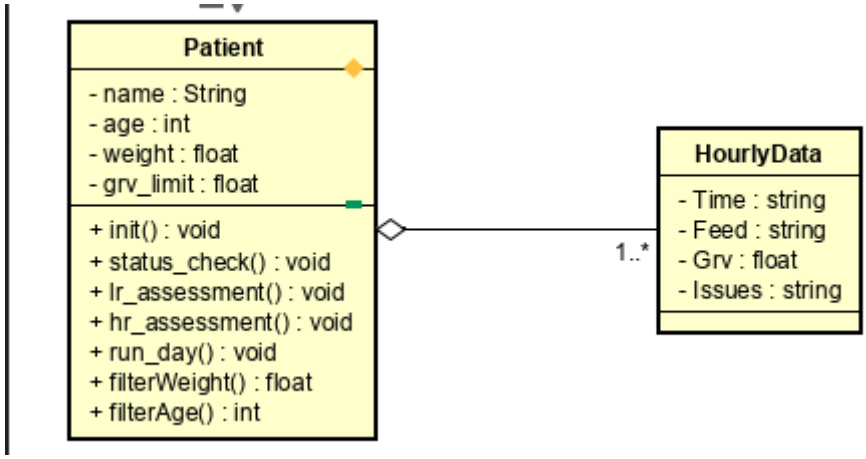
DASDA Assignment 2 - Task 1

Introduction - Design Ideas

Looking at the format of the data we have been provided with. I can see that each patient has the same data attributes across all of them. Based on this, I deemed that it would be appropriate to use classes and objects in order to structure the data I am working with. Using this method of structuring data, it should allows me to access any specific element of the data that I maybe required to use. I will also be incorporating object lists in order to easily loop through every instance of an object easily and display and update all information that is required. In order to read in and assign all of the data from the csv files to appropriate places, I'm planning on creating two classes, the first being for patients and the second being for the per hour data. One of the attributes for the patient class will be an instance of the per hour data class. This lets me access a specific hour of a particular patient with ease, something I thought would be extremely useful whilst planning how to combat the assignment, for testing purposes specifically.

For the algorithm I am using in order to sort the patients by their performance, I am opting to use selection sort as my algorithm of choice. I will give each patient a score as the attribute in which I sort them by. A patients score will increase if they have an Issue at the end of the day, for example; their feeding being stopped or them being referred to a dietician. Therefore at the end of the 5 day period, patients will be sorted accordingly, with the patient with no issues at the top of the list, and the patient who has add the most issues at the bottom. Selection sort works by repeatedly finding the minimum element from the unsorted part of the list and moving it to the beginning of the list. The number of steps selection sort requires is equal to the number of elements in the list squared meaning the more items in the list the longer it will take. Since I am working with a low number of patients, I decided this was an appropriate choice.

Class Diagram



Pseudocode

Reading in CSV files

```
create class instance
create dayClock list

for (til end of csv)
    1st row values are assigned to (patient_number, status, age, weight,
grv_limit)
    4th row till end of file assigned to (time, feed, grv, issues)
    add to dayClock

assign values to class instance of patient
add to patients list
```

Converting Weight from String to Float

```
replace "Weight" with ''
replace "KG" with ''
convert from string to float
return value
```

Converting age from String to Integer

```
replace "Age" with ''
replace "Days" with ''
if "Years" is present
    replace Years with ''
    convert from string to int
    multiply by 365
else
    convert from string to int
return value
```

Check Patient Status

```
if status is equal to LR
    call low risk function
if status is equal to HR
    call high risk function
```

Low risk assessment

```
if day == 1
    call run day function for 1st day
if day  == 2
    call run day function for 2nd day
if day == 3
    call run day function for 3rd day
if day == 4
    call run day function for 4th day
if day == 5
    call run day function for 5th day
```

High risk assessment

```
create int variable
if day less than 4
    while int is less than 72
        patient issues for the first 3 days are NONE
if day is 4
    call low risk assessment
if day is 5
    call low risk assessment
```

Day Simulation

```
create int for hour count
while its less then final hour in day
    if grv is present as number in column
        if its above the limit
            feed becomes 0
            Issue becomes stop feeding
            number of times patient stop fed increased by 1
            if number of times patient feed stop is greater than 2
                patient final issue for that day = REFER TO DIETICIAN
            else
                patient final issue for that day = STOP FEED
        else
            if weight < 40
                feed = 10ML
            else
                feed = 30ML
    if issues are not equal to STOP FEED or REFER TO DIETICIAN
        issues are none
    increase hour count
if final issue for the day is STOP FEED or REFER TO DIETICIAN
    patient score += 1
```

Day End Function

```
print End of (day number)
if day number <= 4
    for i in list
        call get issue function
if day number == 5
    call final_day function
```

Get Issue Function

```
if day number is equal to 1
    print patient name + patient day 1 final issues
if day number is equal to 2
    print patient name + patient days 1-2 final issues
if day number is equal to 3
    print patient name + patient days 1-3 final issues
if day number is equal to 4
    print patient name + patient day 1-4 final issues
```

Final Day Function

```
call sorting algorithm function
print out day 1-5 issues for each patients in order of least problems to most
problems
print see patient info or just exit?
if input is see patient info
    input patient id
    input day number
        call patient info function
```

Patient Info Function

```
check what patient ID and what days info is requested
print patient's name + day
print patient data for that day
```

Selection Sort Algorithm

```
for i in range of the length of the list
    minimum value = i
    for j in range i + 1 of the length of the list
        compare two patients score and select the smallest
    place it at the front of the sorted end of the array
return sorted list
```

Run Function

```
print welcome to PICU system
x = 1
while x < 6
    for each patient in the list of patient
        call check status function
    call day end function
    x + 1
```

Initial Boot Function

```
create patient list
create patient id for each patient
call the csv import function for each patient
call the run function
```