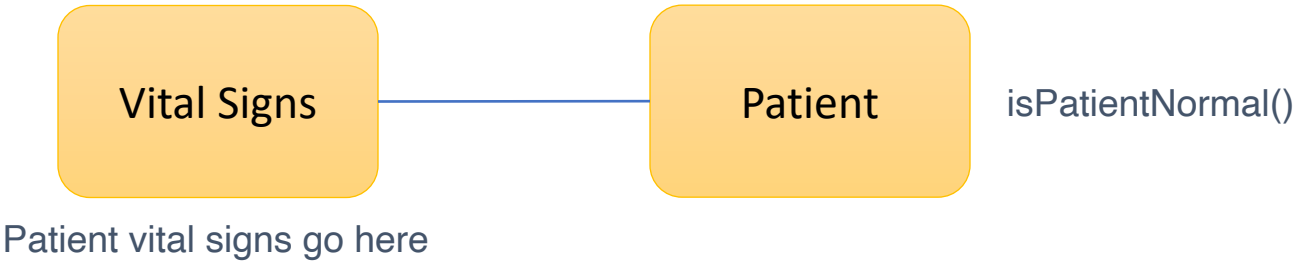
Assignment
Patient Vital Signs

Part 1

Consider the model below and the vital signs table on one of the following slides.

Define a method on the patient class called isPatientNormal(). The method returns true or false based on the rules in the vital signs table. Your job is to define two java classes with all necessary attributes to fulfill the requirements of the rules on the vital signs table.

To test your java implementation, show how to instantiate instances to test each of the six cases presented in the table. Testing your rules by creating test cases for multiple patients in different age groups; assign them vital sign values in all the ranges and use the System.out.println to confirm that your program is working correctly. Also, demonstrate abnormal instances as well.



Pediatric Vital Sign Normal Ranges

Age Group	Respiratory Rate	Heart Rate	Systolic Blood Pressure	Weight in kilos	Weight in pounds
Newborn	30 - 50	120 - 160	50 - 70	2 - 3	4.5 - 7
Infant (1-12 months)	20 - 30	80 - 140	70 - 100	4 - 10	9 - 22
Toddler (1-3 yrs.)	20 - 30	80 - 130	80 - 110	10 - 14	22 - 31
Preschooler (3-5 yrs.)	20 - 30	80 - 120	80 - 110	14 - 18	31 - 40
School Age (6-12 yrs.)	20 - 30	70 - 110	80 - 120	20 - 42	41 - 92
Adolescent (13+ yrs.)	12 - 20	55 - 105	110 - 120	>50	>110

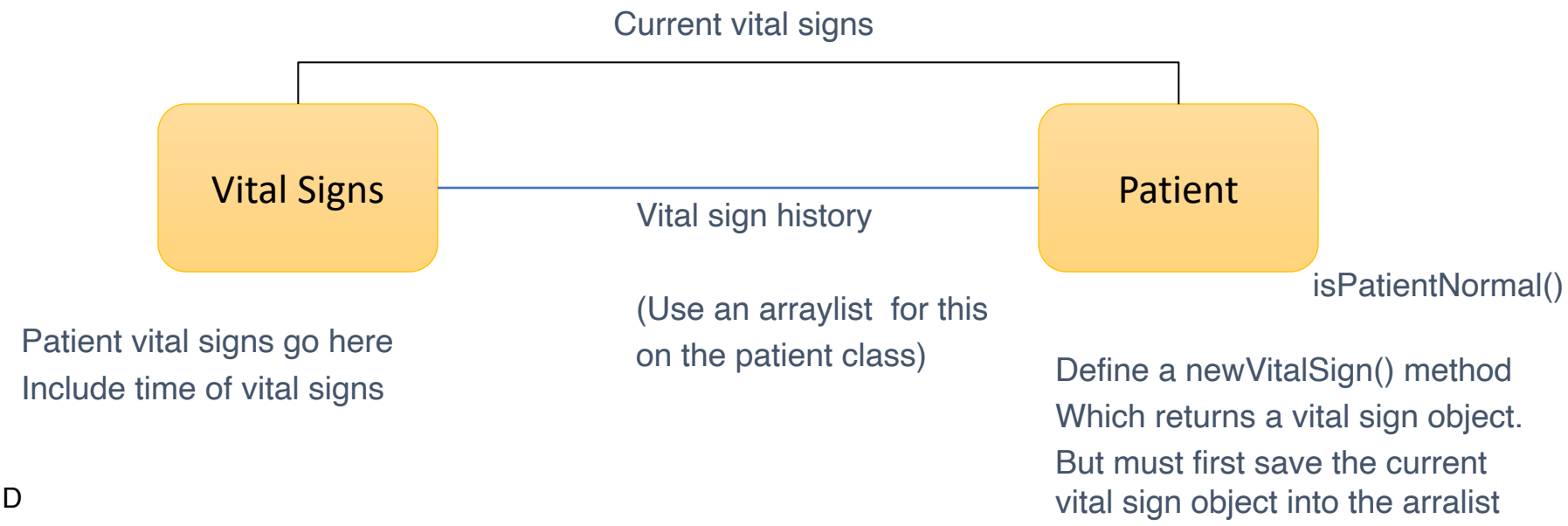
REMEMBER:

- The **patient's** normal range should always be taken into consideration.
- Heart rate, BP & respiratory rate are expected to increase during times of fever or stress.
- Respiratory rate on infants should be counted for a full 60 seconds.
- In a clinically decompensating child, the blood pressure will be the **last** to change. Just because your pediatric patient's BP is normal, don't assume that your patient is "stable".
- Bradycardia in children is an ominous sign, usually a result of hypoxia. Act quickly, as this child is extremely critical.

Kal Bugrara, Ph.D

Part 2

Every time a patient goes for a checkup, the vital signs are captured. That means we must keep a history of all the vital signs over time. In that case we need to keep track of the latest vital sign by using a reference to the latest. When a new vital sign is captured we create a new vital sign object and make it the current vital sign object. But before we do that we must save (move) the current vital (previous) to the vital sign history list (implemented as an array list separate from the current vital sign). Develop a complete java implementation. Create a simple main class to test your application for all possible cases. In addition, print the vital signs history that must include the patient full name, vital signs 1, vital signs 2, etc.



Part 2 (cont.)

Suppose we want to add a new method called isThisVitalSignNormal(vsign). It takes a string which is the string name for one of the vital sign signs and returns true or false. In other words, we give the attribute name as input and the method match that the input to the defined attribute. If match is found then it will check if the range for that attribute is normal. It returns true if yes. Otherwise it returns false. So your method must take the input string and use the input.equals("AttributeName"). If yes then check the normal range.