Class Name: **Date**
Method Signature: **public boolean isValid()**

Test Case #	Requirement	Test Description / Input Data	Expected Result / Output
1	The method will return false for today's date	- Create an instance of Date with the current year, month, and date - Test Input: "9/17/2024"	false
2	The method will return false for any day before today	- Create an instance of Date with the current year and month and a day before the current day - Test Input: "9/16/2024"	false
3	The method will return false for any weekend	- Create an instance of Date that lies on a weekend - Test Input: "2/15/2025"	false
4	The method will return false for any date that is more than 6 months ahead of today	- Create an instance of Date that lies more than 6 months ahead of today - Test Input: "5/10/2026"	false
5	The method will return true if the date is valid and passes all checks	- Create an instance of Date that is valid and passes all checks - Test Input: "10/31/2024"	true
6	"	" - Test Input: "11/28/2024"	true

Class Name: **Profile**
Method Signature: **public int compareTo()**

Test Case #	Requirement	Test Description / Input Data	Expected Result / Output
1	The method will return -1 if the first names and dob are the same, but the profile being compared with has a later alphabetical last name	- Create 2 instances of Profile with both having the same first name and dob but different last names - profile1 = (“John”, “Anderson”, dob1) - profile2 = (“John”, “Brown”, dob1)	-1
2	The method will return -1 if the last names and dob are the same, but the profile being compared with has a later alphabetical first name	- Create 2 instances of Profile with both having the same last names and dob but different first names - profile1 = (“Alice”, “Doe”, dob1) - profile2 = (“Bob”, “Doe”, dob1)	-1
3	The method will return -1 if the first and last names are the same, but the profile being compared with has a later dob	- Create 2 instances of Profile with both having the same names but different dob - profile1 = (“John”, “Doe”, dob1) - profile2 = (“John”, “Doe”, dob2)	-1
4	The method will return -1 if the first names and dob are the same, but the original/first profile has a later alphabetical last name	- Create 2 instances of Profile with both having the same first name and dob but different last names - profile1 = (“John”, “Smith”, dob1) - profile2 = (“John”, “Doe”, dob1)	1
5	The method will return 1 if the last names and dob are the same, but the original/first profile	- Create 2 instances of Profile with both having the same last names and dob but different first names	1

	has a later alphabetical first name	- profile1 = (“Charlie”, “Doe”, dob1) - profile2 = (“Bob”, “Doe”, dob1)	
6	The method will return 1 if the first and last names are the same, but the original/first profile has a later dob	- Create 2 instances of Profile with both having the same names but different dob - profile1 = (“John”, “Doe”, dob3) - profile2 = (“John”, “Doe”, dob1)	1
7	The method will return 0 if both profiles have the same first/last names and dob	- Create 2 instances of Profile with both having the same names and dob - profile1 = (“John”, “Doe”, dob1) - profile2 = (“John”, “Doe”, dob1)	0

Class Name: **List**
Method Signature: **public void add(E e)**

Test Case #	Requirement	Test Description / Input Data	Expected Result / Output
1	The method will successfully add a doctor to the provider list if the doctor object is successfully created, meaning a valid profile, location, specialty, and npi #.	- Create 1 instance of Doctor with the same information as the first doctor in providers.txt - Doctor doctor1 = new Doctor(profile1, location1, specialty1, "01"); -providerList.add(doctor1);	doctor1 is added to the list, prints “Test 1 - Add Doctor: Passed”
2	The method will successfully add a technician list if the technician object is successfully created, meaning a valid profile, location, and charging rate.	- Create 1 instance of Technician with the same information as the first technician in providers.txt - Technician technician1 = new Technician(profile2,	Technician1 is added to the list, prints “Test 2 - Add Technician: Passed”

		location2, 110); -providerList.add(technician1);	

Class Name: **List**
Method Signature: **public void remove(E e)**

Test Case #	Requirement	Test Description / Input Data	Expected Result / Output
1	The method will successfully remove a doctor from the provider list if the doctor object exists in the list.	- Call the method to remove doctor1 from the list providerList.remove(doctor1);	doctor1 is removed from the list, prints “Test 3 - Remove Doctor: Passed”
2	.The method will successfully remove a technician from the provider list if the technician object exists in the list.	- Call the method to remove technician1 from the list providerList.remove(technician1);	technicial1 is removed from the list, prints “Test 4 - Remove Technician: Passed”