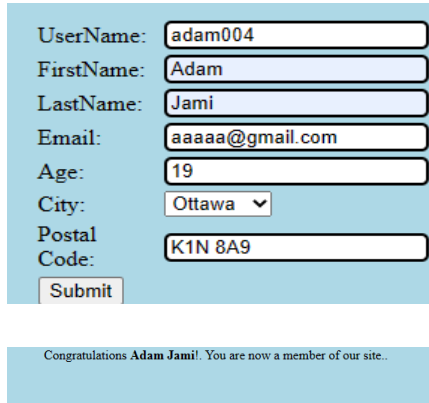



SEG 3503 Lab 2
Adam Jami
300245165
ajami070@uottawa.ca

Github

[adamjami2004/seg3503_playground \(github.com\)](https://github.com/adamjami2004/seg3503_playground)

Exercice 1:

Cas de test	résultats escomptés	résultats actuels	Verdict
1	accepté	<p>accepté</p> 	Pass
2	error 4	<p>error 4</p> 	Fail

3	error 1 4	error 1 4 5 <div> UserName: <input type="text" value="aaa"/> FirstName: <input type="text"/> LastName: <input type="text"/> Email: <input type="text"/> Age: <input type="text" value="2"/> City: <input type="text" value="Halifax"/> Postal Code: <input type="text" value="K1N 8A9"/> <input type="button" value="Submit"/> </div> <div> UserName: <input type="text" value="aaa"/> <small>Size of UserName must be between 6 and 12</small> FirstName: <input type="text"/> LastName: <input type="text"/> Email: <input type="text"/> <small>Wrong Email format</small> Age: <input type="text" value="2"/> <small>doit être au minimum égal a 18</small> City: <input type="text" value="Halifax"/> Postal Code: <input type="text" value="K1N 8A9"/> <input type="button" value="Submit"/> </div>	Failed
4	accepté	accepté <div> UserName: <input type="text" value="adamdakd"/> FirstName: <input type="text"/> LastName: <input type="text"/> Email: <input type="text" value="ajami070@uottawa.ca"/> Age: <input type="text" value="21"/> City: <input type="text" value="Ottawa"/> Postal Code: <input type="text" value="K1N 8A9"/> <input type="button" value="Submit"/> </div> <div> Congratulations !. You are now a member of our site.. </div>	Pass
5	error 7	error 7 <div> UserName: <input type="text" value="adamdakd"/> FirstName: <input type="text"/> LastName: <input type="text"/> Email: <input type="text" value="ajami070@uottawa.ca"/> Age: <input type="text" value="21"/> City: <input type="text" value="Halifax"/> Postal Code: <input type="text" value="FRf"/> <input type="button" value="Submit"/> </div>	Failed

		<div><div>UserName: <input type="text" value="adamdakd"/></div><div>FirstName: <input type="text"/></div><div>LastName: <input type="text"/></div><div>Email: <input type="text" value="ajami070@uottawa.ca"/></div><div>Age: <input type="text" value="21"/></div><div>City: <input type="text" value="Halifax"/></div><div>Postal Code: <input type="text" value="FRF"/> Wrong Postal Code format</div><div><input type="button" value="Submit"/></div></div>	
6	accepté	<div>accepté</div> <div><div>UserName: <input type="text" value="qaaaaaaaaa"/></div><div>FirstName: <input type="text" value="aaaaaaaaaaaa"/></div><div>LastName: <input type="text" value="aaaaaaaaaaaa"/></div><div>Email: <input type="text" value="ajami070@uottawa.ca"/></div><div>Age: <input type="text" value="18"/></div><div>City: <input type="text" value="Halifax"/></div><div>Postal Code: <input type="text" value="K1N 8A7"/></div><div><input type="button" value="Submit"/></div></div> <div><div>Congratulations aaaaaaaaaa aaaaaaaaaa! You are now a member of our site..</div></div>	Passed

Exercice 2

Output avant de programmer les tests

```
$ bin/test

Thanks for using JUnit! Support its de

?
?? JUnit Jupiter ?
? ?? DateTest ?
? ? ?? nextDate_sample() ?
? ?? BitTest ?
? ?? constructor_int_ok() ?
? ?? constructor_int_tooLarge() ?
? ?? constructor_int_tooSmall() ?
? ?? constructor_Bit() ?
? ?? hashCode_values() ?
? ?? getIntValue() ?
? ?? equals() ?
? ?? toString_values() ?
? ?? or() ?
? ?? and() ?
? ?? not() ?
? ?? xor() ?
? ?? setValue() ?
? ?? constructor_default_0() ?
?? JUnit Vintage ?
?? BitAndTest ?
?? [0] ?
? ?? testAnd[0] ?
?? [1] ?
? ?? testAnd[1] ?
?? [2] ?
? ?? testAnd[2] ?
?? [3] ?
?? testAnd[3] ?

Test run finished after 103 ms
[      9 containers found      ]
[      0 containers skipped    ]
[      9 containers started    ]
[      0 containers aborted    ]
[      9 containers successful ]
[      0 containers failed     ]
[     19 tests found           ]
[      0 tests skipped         ]
[     19 tests started         ]
[      0 tests aborted         ]
[     19 tests successful      ]
[      0 tests failed          ]
```

bin/test :

```
Test run finished after 125 ms
[      31 containers found      ]
[      0 containers skipped     ]
[      31 containers started    ]
[      0 containers aborted     ]
[      31 containers successful ]
[      0 containers failed      ]
[      58 tests found           ]
[      0 tests skipped          ]
[      58 tests started         ]
[      0 tests aborted          ]
[      58 tests successful      ]
[      0 tests failed           ]
```

typical explicit test that does not use exceptions :

```
@Test
void nextDate_sample8() {
    Date d = new Date(1800,11,29);
    assertEquals(new Date(1800,11,30), d.nextDate());
}
```

explicit cases that have exceptions:

```
@Test
void nextDate_sample16() {
    assertThrows(IllegalArgumentException.class, () -> new Date(1550, 2, 31).nextDate(), "Jour doit etre au maximum 28 car année non bissextile.");
}

@Test
void nextDate_sample17() {
    assertThrows(IllegalArgumentException.class, () -> new Date(1550, 2, 29).nextDate(), "Jour doit etre au maximum 28 car année non bissextile.");
}

@Test
void nextDate_sample18() {
    assertThrows(IllegalArgumentException.class, () -> new Date(-1, 10, 20).nextDate(), "L'année doit etre supérieure ou égale à 0.");
}

@Test
void nextDate_sample19() {
    assertThrows(IllegalArgumentException.class, () -> new Date(1458, 15, 12).nextDate(), "Le mois doit etre entre 1 et 12. ");
}

@Test
void nextDate_sample20() {
    assertThrows(IllegalArgumentException.class, () -> new Date(1975, 6, -50).nextDate(), "Jour doit être entre 1 et 31.");
}
```

Parameterized tests for values that have an actual value :

```
List<Integer[]> params = new LinkedList<Integer[]>( );
params.add(new Integer[] {1700,06,20,1700,06,21});
params.add(new Integer[] {2005,4,15,2005,4,16});
params.add(new Integer[] {1901,7,20,1901,7,21});
params.add(new Integer[] {3456,3,27,3456,3,28});
params.add(new Integer[] {1500,2,17,1500,2,18});
params.add(new Integer[] {1700,6,29,1700,6,30});
params.add(new Integer[] {1800,11,29,1800,11,30});
params.add(new Integer[] {3453,1,29,3453,1,30});
params.add(new Integer[] {444,2,29,444,3,1});
params.add(new Integer[] {2005,4,30,2005,5,1});
params.add(new Integer[] {3453,1,30,3453,1,31});
params.add(new Integer[] {3456,3,30,3456,3,31});
params.add(new Integer[] {1901,7,31,1901,8,1});
params.add(new Integer[] {3453,1,31,3453,2,1});
params.add(new Integer[] {3456,12,31,3457,1,1});
return params;
```

Parameterized tests for values that do result in an exception:

```
public static List<Integer[]> data() {  
    List<Integer[]> params = new LinkedList<Integer[]>();  
    params.add(new Integer[] { 1500, 2, 31 });  
    params.add(new Integer[] { 1500, 2, 29 });  
    params.add(new Integer[] { -1, 10, 20 });  
    params.add(new Integer[] { -1458, 15, 12 });  
    params.add(new Integer[] { 1975, 6, -50 });  
    return params;  
}
```