Testing Report for Individual Prototype (Week 7)

> All tests are conducted in sequential order by **WorldTest**

Test 1: Creating a new World object

Purpose	Input	Expected Output	Actual Output/Result
World constructor and initialisation [illegal case]	Ground array of wrong dimensions	No error messages (there is a precondition)	No error messages
World constructor and initialisation [normal case]	Ground array of correct dimensions	No error messages	No error messages
World.getLifeForms() [boundary case]	No life forms have been added to the world	size() = 0	As expected
World.getLifeForms() [normal case]	6 penguins and 5 fish have been added to the world	size() = 11	As expected

Test 2: LifeForm's accelerate() and move() methods

Purpose	Input	Expected Output	Actual Output/Result
accelerate() and move() [normal case with *positive* acceleration: there are no illegal cases, all inputs accepted]	Acceleration of (1.0, 1.0, 0.5), Move with zero current, Penguin initialised in the middle of the world (5.0, 5.0, 5.0) with no velocity	*Velocity*: x=0.0666…, y=0.0666…, z=0.0333… Position: x=5.0666…, y=5.0666, z=5.0333… *Status*: PENGUIN_STATUS_RELAXING	All values as expected. Speed = √(0.0666^2 + 0.0666^2 + 0.0333^2) = 0.01, which is the maximum speed defined for a penguin (as defined in the Penguin class)
accelerate() and move() [normal case with *negative* acceleration]	Acceleration of (-1.0, -1.0, -0.5), Move with zero current, Penguin continues from its current position	*Velocity*: x=-0.0666…, y= -0.0666…, z=-0.0333… Position: x=5.0, y=5.0, z=5.0 *Status*: PENGUIN_STATUS_RELAXING	All values as expected. The speed is, once again, less than or equal to the maximum speed for a penguin. It also ends up where it started.

Test 3: Testing the Penguin's ability to move itself into PENGUIN_STATUS_PORPOISING (a mode where it goes to the surface to get air) when the oxygen level is low, and then die of old age

Purpose	Input	Expected Output	Actual Output/Result
Penguin.tick() [Normal case: there are no illegal cases]	A new Penguin in the middle of the world (5.0, 5.0, 5.0) with full (1.0) oxygen	When the Penguin first enters Porpoising mode, its oxygen level should be just under the threshold. The first porpoising point is on the surface randomly placed near the middle, and the second point is the reflection of the position about the middle. The Penguin will self-destruct after 720 ticks by sending an ACTION_KILL signal.	(The black text is copied from the console) Oxygen level: 0.599975219046598 < 0.6 as expected. Penguin reached status PORPOISING at 397 Porpoising Information: Position: Vector3D[x=5.0,y=5.0,z=5.0] Point 1: Vector3D[x=5.4620350774874575,y=0.0, z=5.469817314697662] Point 2: Vector3D[x=5.0,y=5.0,z=5.0] The first point is within limits and the second has been calculated correctly. Age: 399 After 720 ticks, the action instruction returned the world is: ACTION_KILL As expected.

Test 4: To investigate the world's ability until death and porpoising with movement in the world

Purpose	Input	Expected Output
`World.tick()` [Normal case: there are no illegal cases]	A new world was instantiated to provide a clean slate to work off. A new `Penguin` in the middle of the world (5.0, 5.0, 5.0) with full (1.0) oxygen	The `Penguin` will move around randomly until its oxygen falls below threshold level. Then it will move consistently towards Point 1, then towards Point 2, and then randomly once porpoising has concluded. It should be dead after 720 ticks.

Actual Output/Result

(The black text is copied from the console; numbers may change due to random nature)

```
i = 0
Vector3D[x=4.996080358954351,y=5.00722959697474,z=4.998802411664147]
…
Oxygen level: 0.5996429847293048
Penguin reached status PORPOISING at 397
Porpoising Information:
Position: Vector3D[x=9.411819001969901,y=4.335821324500113,z=1.57869445567968]
Point 1: Vector3D[x=4.547481116891608,y=0.0,z=4.623218873492785]
Point 2: Vector3D[x=0.5881809980300989,y=5.664178675499887,z=8.42130554432032]
Age: 399
```

Porpoising information is consistent.

```
i = 400
Vector3D[x=9.187112073734308,y=4.491191680287328,z=1.6850118183700933]
Information: Penguin reached y = 0 at age 475
i = 500
Vector3D[x=2.9316246945115836,y=1.402350742385981,z=6.052544443218176]
Information: Porpoising Concluded at age 547
…
i = 700
Vector3D[x=6.384767010153499,y=4.220553560354408,z=7.387602757622916]
Movement is completely random after porpoising,
After 720 ticks, the number of life forms remaining in the world is: 0
```

As expected

Test 5: `Vector3D` class

Note that there are no operations on Vector3D objects with illegal inputs. All inputs within range are accepted.

Purpose	Input	Expected Output	Actual Output/Result
`equals()` [Normal true case]	Comparing (1,2,3) with (1,2,3)	True	As expected
`equals()` [Normal false case]	Comparing (1,2,3) with (0,0,0)	False	As expected
`equals()` [Exceptional case]	Comparing (1,2,3) with null	False	As expected
`multiply()`[Normal positive case]	Multiply (1,2,3) by 0.5	Vector3D[x=0.5,y=1.0,z=1.5]	As expected
`multiply()`[Boundary zero case]	Multiply (1,2,3) by 0	Vector3D[x=0.0,y=0.0,z=0.0]	As expected
`multiply()`[Normal negative case]	Multiply (1,2,3) by -0.5	Vector3D[x=-0.5,y=-1.0,z=-1.5]	As expected
`add()` [Boundary zero case]	Add (1,2,3) with (0,0,0)	Vector3D[x=1.0,y=2.0,z=3.0]	As expected
`add()` [Normal case]	Add (1,2,3) with (-1,-2,-3)	Vector3D[x=0.0,y=0.0,z=0.0]	As expected
`length()` [Normal case]	Length of (1,2,3)	3.7416573…	As expected
`length()` [Boundary case]	Length of (0,0,0)	0	As expected
`unitVector()` [Normal case]	Unit vector of (1,2,3)	Vector3D[x=0.2672612…,y=0.5345224…,z=0.8017837…]	As expected
`unitVector()` [Special case] *	Unit vector of (0,0,0)	Vector3D[x=0.0,y=0.0,z=0.0]	As expected

* Not actually mathematically defined – treated as the zero vector for computational convenience