

LAPORAN TUGAS BESAR 2

IF2210/Pemrograman Berorientasi Objek

ArkavQuarium

Dipersiapkan oleh:

K02M - Deny in Spanish

13515020 - Daniel Christian Pradipta Baso


13515116 - Aries Tri Sutrisno K A

13516026 - William Juniarta Hadiman

13516041 - Felix Septianus Darmawan

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2210-TB-K02M-2</i>		27
		<i>Revisi</i>	0	2018-04-25

STEI- ITB	<i>IF2210-TB-K02M-2</i>	Halaman 1 dari 27 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

Daftar Isi

Ringkasan	4
Penjelasan Tambahan Spesifikasi Tugas	4
Background	4
Main Menu	4
Penentuan Posisi Makanan Ikan Dilakukan Menggunakan Mouse	4
Pemain Dapat Mengambil Coin Menggunakan Mouse	4
Ikan Memiliki Tampilan Berbeda Saat Lapar	4
Rancangan Kelas	4
Perubahan dari Tugas Besar	10
Rincian Kelas	10
Aquarium	10
Vector2	10
AquariumObject	11
Creature	11
Fish	12
IDestructible	12
Guppy	12
Piranha	13
Snail	13
Coin	13
Food	13
Element List	14
Linked List	14
Program Utama	14
Test Script	14
Pengukuran Metriks Aplikasi	21
Pengukuran Kualitas Kode Aplikasi	22
Pembagian Kerja dalam Kelompok	22
Lampiran	23

Form Asistensi	23
Log Activity Anggota Kelompok	24
Screenshot Program	25

1 Ringkasan

Dalam tugas besar ini kami diminta untuk mengimplementasikan permainan Insaniquarium dengan nama ArkavQuarium, dimana Insaniquarium adalah sebuah game di mana kita dapat memelihara ikan yang akan menghasilkan uang yang dapat kita kumpulkan. Terdapat beberapa ikan dan hewan lain yang dapat dipelihara.

Laporan yang kami buat berisi tentang spesifikasi tugas tambahan yang kami tambahkan dengan penjelasan tiap spesifikasi, lalu ada rancangan kelas yang kami buat berisi Diagram Kelasnya dan juga rincian isi tiap kelasnya. Selanjutnya kami memberikan penjelasan mengenai program utama yang kami buat lalu yang terakhir adalah pengetesan program yang kami lakukan.

Hasil tugas besar ini sangat memuaskan. Dengan desain kelas pada tugas kecil yang baik, implementasi dan interaksi antar kelas menjadi sangat mudah dan rapi. Tugas besar ini sangat baik untuk mengaplikasikan pemrograman berorientasi objek.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 Background

Akuarium dapat mencetak gambar background.

2.2 Main Menu

Memunculkan tombol “Play Game” di awal permainan. Pemain akan menekan tombol tersebut untuk masuk ke game.

2.3 Penentuan Posisi Makanan Ikan Dilakukan Menggunakan Mouse

Posisi X dari kursor akan menentukan posisi X makanan muncul ketika tombol mouse ditekan, namun posisi Y tetap sama dengan nol.

2.4 Pemain Dapat Mengambil Coin Menggunakan Mouse

Coin dapat diambil oleh pemain jika pemain menekan mouse dan posisi mouse berada dalam jarak 16 pixel dari tengah koin.

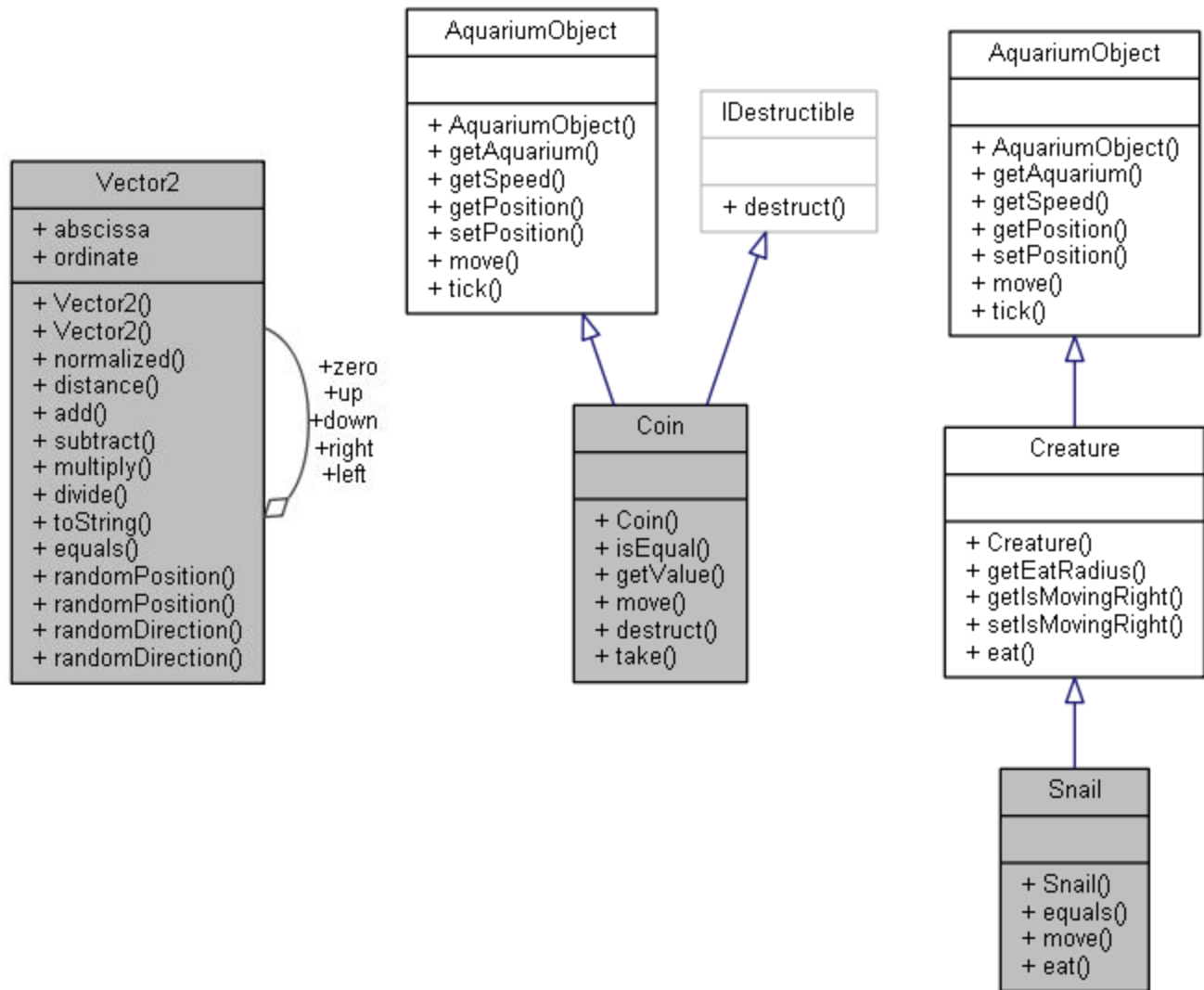
2.5 Ikan Memiliki Tampilan Berbeda Saat Lapar

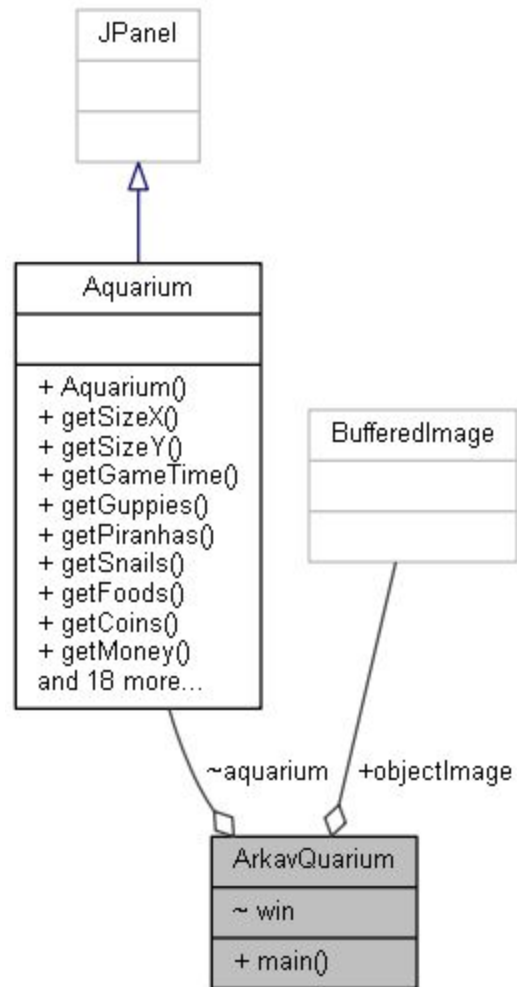
Ikan yang sedang dalam keadaan lapar akan memiliki tampilan yang berbeda.

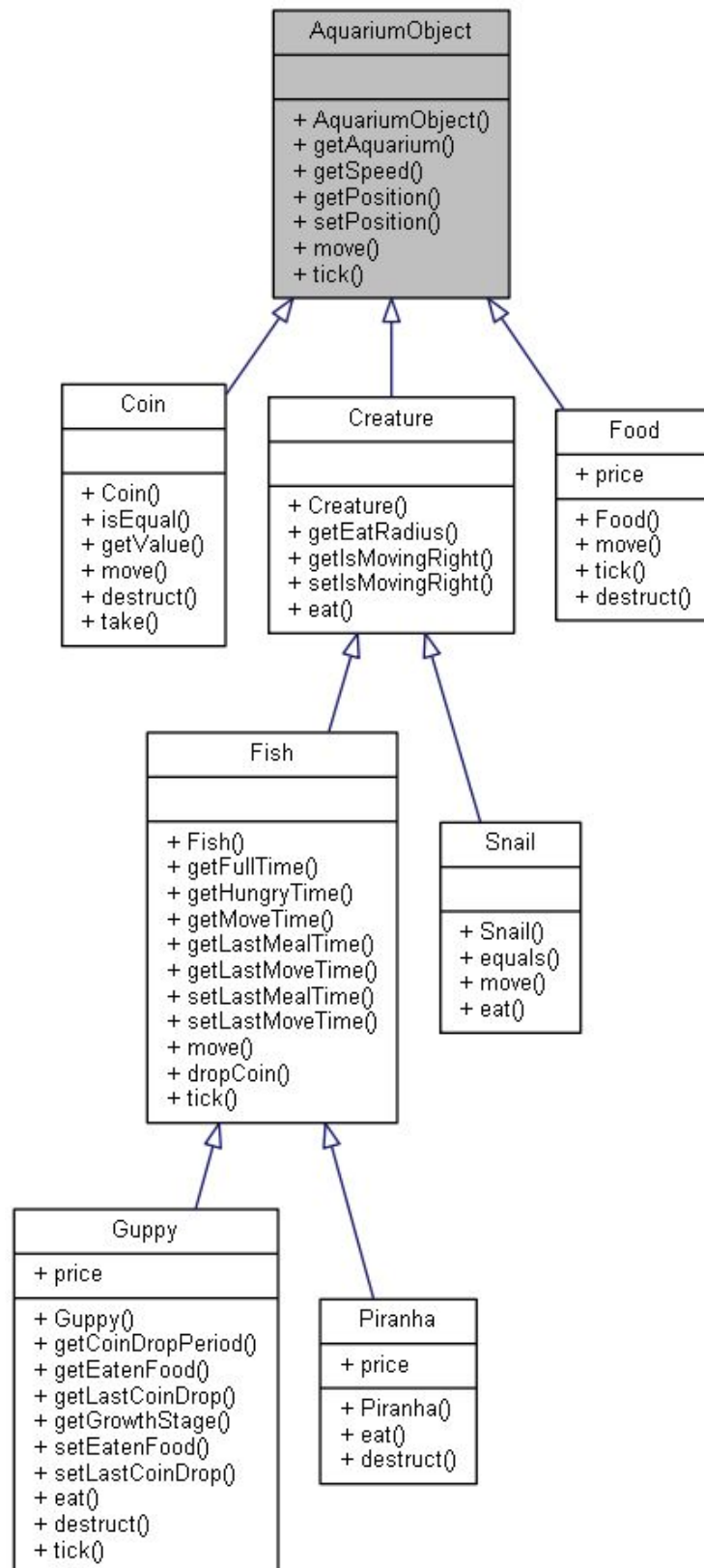
3 Rancangan Kelas

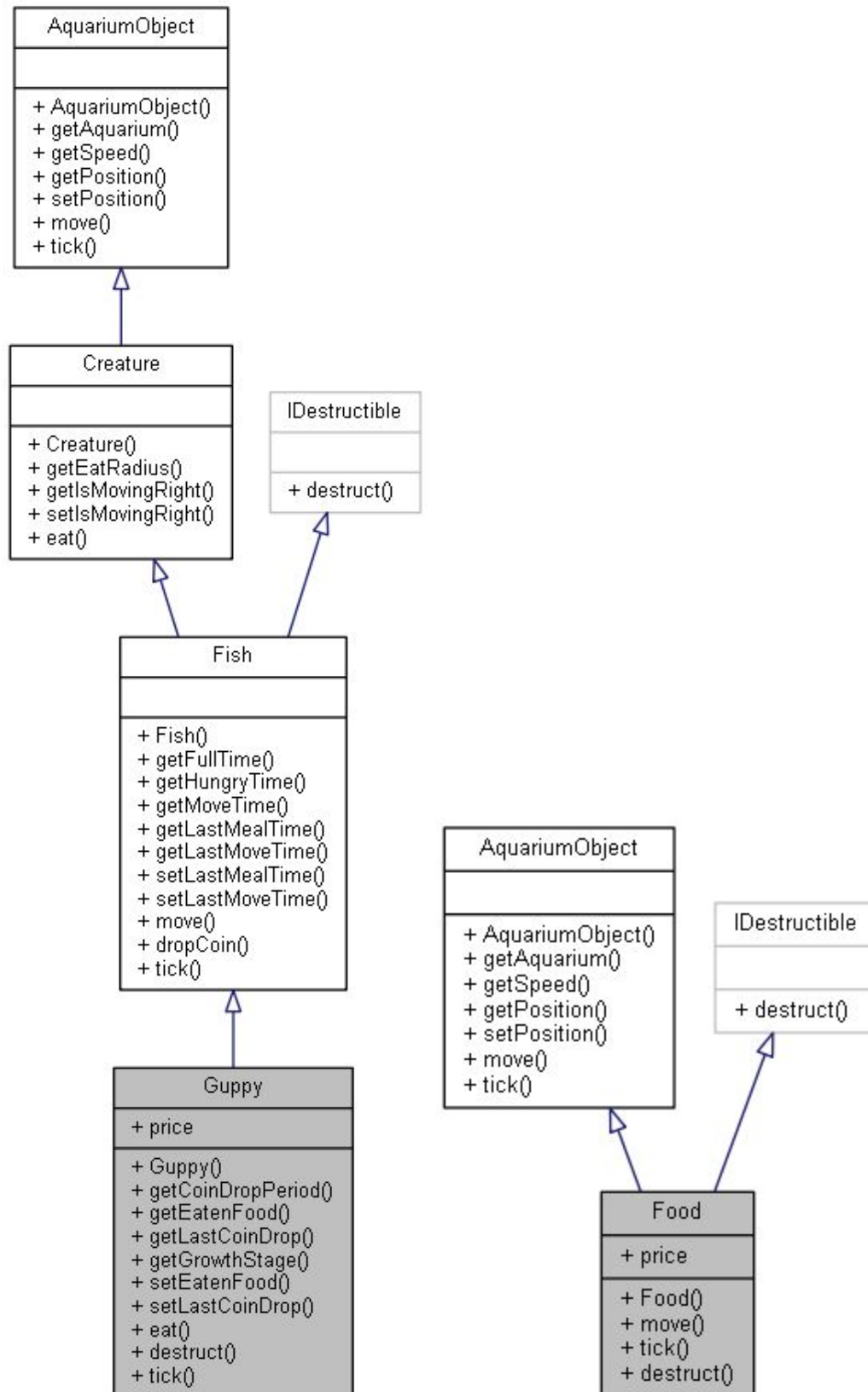
Rancangan kelas yang kami buat berisi 1 interface dan 13 kelas. Interface tersebut bernama kelas IDestructible. Sebelas kelas tersebut adalah Aquarium, AquariumObject, Coin, Creature, Fish, Food, Guppy, LinkedList, ElementList, Piranha, Snail, dan Vector2. Dari sebelas kelas tersebut,

kelas yang tidak abstrak hanyalah Vector2, ElementList, Aquarium, Coin, Food, Guppy, Piranha, dan Snail. LinkedList adalah kelas generic yang berfungsi sebagai penyimpanan data.









3.1 Perubahan dari Tugas Besar

C++	Java
Aquarium bukan turunan dari kelas apapun	Aquarium turunan dari JPanel
Tick menggunakan infinite loop dibatasi waktu	Tick menggunakan timer

4 Rincian Kelas

4.1 Aquarium

Kelas Aquarium berfungsi sebagai pengontrol waktu game dan penyimpanan objek yang ada di dalam game.

Method:

- Constructor: Membuat objek
- Getter: Mengembalikan nilai variabel
- Setter: Mengubah nilai variabel
- Add: Menambahkan objek ke akuarium
- Remove: Menghapus objek dari akuarium
- Tick: Menambah gameTime sebanyak satu dan memanggil tick dari semua objek yang ada di akuarium
- DrawComponent: Mencetak akuarium ke window

Atribut:

- SizeX: Lebar akuarium
- SizeY: Tinggi akuarium
- GameTime: Unit waktu dari game
- Guppies: Penyimpanan guppy
- Piranhas: Penyimpanan piranha
- Snails: Penyimpanan snail
- Foods: Penyimpanan food
- Coins: Penyimpanan coin
- Money: Jumlah uang yang dipakai untuk membeli
- Egg: Jumlah telur yang telah dibeli
- EggPrice: Harga telur

4.2 Vector2

Kelas Vector2 berfungsi sebagai tipe bentukan untuk posisi dan arah.

Method:

- Constructor: Membuat objek
- Getter: Mengembalikan nilai variabel
- Setter: Mengubah nilai variabel
- IsEqual: Membandingkan kesamaan dengan Vector2 lain
- Add: Menambahkan nilai x dan y dengan Vector2 lain
- Subtract: Mengurangi nilai x dan y dengan Vector2 lain
- Multiply: Mengalikan nilai x dan y dengan k
- Divide: Membagi nilai x dan y dengan k
- Normalized: Membuat ukuran vektor sebesar 1 tanpa mengubah arahnya
- Distance: Menghitung jarak ke Vector2 lain
- RandomPosition: Memberikan Vector2 yang memiliki x dalam range 0-x, dan y dalam range 0-y
- RandomDirection: Memberikan Vector2 yang memiliki arah acak dengan ukuran 1
- ToString: Mengembalikan Vector2 dalam bentuk string

Atribut:

- Zero: Mengembalikan Vector2 dengan $x = 0$ dan $y = 0$
- Right: Mengembalikan Vector2 dengan $x = 1$ dan $y = 0$
- Left: Mengembalikan Vector2 dengan $x = -1$ dan $y = 0$
- Up: Mengembalikan Vector2 dengan $x = 0$ dan $y = -1$
- Down: Mengembalikan Vector2 dengan $x = 0$ dan $y = 1$

4.3 AquariumObject

Kelas AquariumObject berfungsi sebagai kelas paling mendasar dari objek yang ada di dalam akuarium

Method:

- Constructor: Membuat objek
- Getter: Mengembalikan nilai variabel
- Setter: Mengubah nilai variabel
- Move: Fungsi abstrak untuk bergerak
- Tick: Fungsi virtual untuk melakukan aksi setiap satu satuan waktu

Atribut:

- Aquarium: Referensi ke akuarium ikan ini berada
- Speed: Kecepatan objek bergerak
- Position: Posisi objek berada

4.4 Creature

Kelas Creature berfungsi sebagai kelas dari objek yang hidup. Kelas ini adalah turunan dari AquariumObject

Method:

- Constructor: Membuat creature
- Getter: Mengembalikan nilai variabel

- Setter: Mengubah nilai variabel
- Eat: Fungsi abstrak untuk makan

Atribut:

- EatRadius: Jarak maksimum creature dapat meraih makanan
- IsMovingRight: Menentukan arah horizontal creature bergerak

4.5 Fish

Kelas Fish berfungsi sebagai kelas dari objek yang dapat menghasilkan coin. Kelas ini adalah turunan dari Creature.

Method:

- Constructor: Membuat fish
- Getter: Mengembalikan nilai variabel
- Setter: Mengubah nilai variabel
- Move: Menggerakkan fish jika lapar dan terdapat makanan, jika kondisi tersebut tidak terpenuhi, maka ikan akan bergerak acak dan berganti arah setiap moveTime
- MoveRandomly: Mengarahkan fish untuk bergerak secara acak
- DropCoin: Membuat coin di posisi fish
- Tick: Menghilangkan fish jika fish lapar melebihi hungryTime

Atribut:

- FullTime: Durasi fish kenyang sebelum lapar
- HungryTime: Durasi fish lapar sebelum mati
- MoveTime: Durasi fish akan bergerak ke arah yang sama sebelum berganti arah
- LastMealTime: Waktu terakhir fish makan
- LastMoveTime: Waktu terakhir fish berganti arah
- Direction: Arah ikan ini bergerak

4.6 IDestructible

Kelas IDestructible berfungsi sebagai kelas dari objek yang dapat dihancurkan dari akuarium. Kelas ini adalah interface.

Method:

- Destruct: Menghilangkan objek dari akuarium

4.7 Guppy

Kelas Guppy berfungsi sebagai kelas dari Guppy. Kelas ini adalah turunan dari Fish.

Method:

- Constructor: Membuat guppy
- Getter: Mengembalikan nilai variabel
- Setter: Mengubah nilai variabel
- GetGrowthStage: Mengembalikan tahap pertumbuhan guppy dari jumlah makanan yang telah dimakan
- Eat: Fungsi untuk memakan food di sekitar guppy
- FindFood: Fungsi untuk mendapatkan lokasi food di akuarium

- Destruct: Menghilangkan guppy dari list of guppies
- Tick: Menjatuhkan koin setiap coinDropPeriod

Atribut:

- CoinDropPeriod: Periode guppy akan menjatuhkan coin
- EatenFood: Jumlah food yang telah dimakan guppy
- LastCoinDrop: Waktu terakhir guppy menjatuhkan coin
- Price: Harga dari guppy

4.8 Piranha

Kelas Piranha berfungsi sebagai kelas dari Piranha. Kelas ini adalah turunan dari Fish.

Method:

- Constructor: Membuat piranha
- Eat: Fungsi untuk memakan guppy di sekitar piranha
- FindFood: Fungsi untuk mendapatkan lokasi guppy di akuarium
- Destruct: Menghilangkan piranha dari list of piranhas

Atribut:

- Price: Harga dari piranha

4.9 Snail

Kelas Snail berfungsi sebagai kelas dari Snail. Kelas ini adalah turunan dari Creature.

Method:

- Constructor: Membuat snail
- Move: Fungsi untuk bergerak horizontal di dasar akuarium
- Eat: Fungsi untuk memakan coin di sekitar snail
- FindFood: Fungsi untuk mendapatkan lokasi koin di akuarium

4.10 Coin

Kelas Coin berfungsi sebagai kelas dari coin. Kelas ini adalah turunan dari AquariumObject.

Method:

- Constructor: Membuat coin
- Getter: Mengembalikan nilai variabel
- Move: Menggerakan coin ke dasar akuarium
- Take: Menambahkan money sebanyak value
- Destruct: Menghilangkan coin dari list of coins

Atribut:

- Value: Nilai dari coin

4.11 Food

Kelas Coin berfungsi sebagai kelas dari coin. Kelas ini adalah turunan dari AquariumObject.

Method:

- Constructor: Membuat food

- Move: Menggerakkan food ke dasar akuarium dan menghancurkan food ketika menyentuh dasar akuarium
- Destruct: Menghilangkan food dari list of foods

Atribut:

- Price: Harga dari food

4.12 Element List

Kelas Element List adalah kelas yang berisi element dari list.

Method:

- Constructor: Membuat element list
- GetData: Mengembalikan data daripada list
- GetNext: Mengembalikan nilai selanjutnya dari element list
- SetNext: Mengubah nilai selanjutnya daripada element list

Atribut:

- Data: variabel penyimpanan data bertipe T (template)
- Next: referensi alamat element list selanjutnya

4.13 Linked List

Kelas Linked List adalah kelas yang mengurus element list yang diurutkan menjadi linked list.

Method:

- Constructor: mengubah variabel first menjadi null
- GetFirst: mengembalikan element list pertama
- Find: mengembalikan index dari elemen dan mengembalikan -1 jika elemen tidak ditemukan
- IsEmpty: mengembalikan true jika dan hanya jika variabel first sama dengan null
- Add: menambah elemen list baru bernilai sama dengan argumen
- Remove: menghapus elemen list pertama yang bernilai sama dengan argumen

5 Program Utama

Program utama akan menginisialisasi gambar-gambar yang dibutuhkan. Lalu, window untuk program dibuat. Selanjutnya, main menu akan ditampilkan dan event klik button play game ditangkap. Jika pemain menekan tombol “Play Game” maka game akan diinisialisasi dengan membuat akuarium dan mengeset waktu. Lalu, main menu akan dihapus dan aquarium dibuat dan ditampilkan ke layar. Kemudian, timer akan dijalankan. Isi dari timer adalah mengecek kondisi menang/kalah dan melakukan tick akuarium. Jika kondisi menang/kalah telah diset, timer akan berhenti.

6 Test Script

STEI- ITB	IF2210-TB-K02M-2	Halaman 14 dari 27 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

No.	Kelas	Nama File Driver	Fitur/Method yang diuji	Kasus Pengujian	Hasil yang Diharapkan	Hasil yang Keluar
1	Aquarium	AquariumTest	getSizeX()	Aquarium(50,50,100,10)	50	50
2	Aquarium	AquariumTest	getSizeY()	Aquarium(50,50,100,10)	50	50
3	Aquarium	AquariumTest	getGameTime()	default, setGameTime(0)	0	0
4	Aquarium	AquariumTest	getGuppies() getPiranhas() getSnails() getCoins() getFoods()	default, linked lists ctor	not null	not null
5	Aquarium	AquariumTest	getMoney()	Aquarium(50,50,100,10)	100	100
6	Aquarium	AquariumTest	getEgg()	default, setEgg(0)	0	0
7	Aquarium	AquariumTest	getEggPrice()	Aquarium(50,50,100,10)	10	10
8	Aquarium	AquariumTest	setGameTime()	setGameTime(192)	192	192
9	Aquarium	AquariumTest	setMoney()	setMoney(168)	168	168
10	Aquarium	AquariumTest	setEgg()	setEgg(11)	11	11
11	Aquarium	AquariumTest	setEggPrice()	setEggPrice(19216811)	19216811	19216811
12	Aquarium	AquariumTest	add(Guppy) add(Piranha) add(Snail) add(Coin) add(Food)	add(guppytest); add(piranhatest); add(snailtest); add(cointest); add(foodtest);	not null not null not null not null not null	not null not null not null not null not null
13	Aquarium	AquariumTest	remove(Guppy) remove(Piranha) remove(Snail) remove(Coin) remove(Food)	add(guppytest); remove(guppytest); add(piranhatest); remove(piranhatest); remove(aqtest2.getSnails(). getFirst().getData()); .add(cointest); remove(cointest); add(foodtest); .remove(foodtest);	null null null null null	null null null null null
14	Aquarium	AquariumTest	tick()	setGameTime(0); tick();	1	1
15	Coin	CoinTest	getValue()	Coin(aqtest,vectest,10);	10	10
16	Coin	CoinTest	isEqual()	test1.isEqual(test1)	true	true
17	Coin	CoinTest	destruct()	aqtest.add(test2);	-1	-1

				test2.destruct(); find(test2);		
18	Coin	CoinTest	take()	int curMoney = aqtest.getMoney(); int coinVal = test1.getValue(); test1.take(); assertEquals(curMoney+co inVal,test1.getAquarium(). getMoney());	110	110
	Creature	CreatureTest	Karena Creature merupakan kelas abstrak, maka dites dengan melakukan instance kelas turunannya (kelas Snail)			
19	Creature	CreatureTest	getEatRadius()	default	50	50
20	Creature	CreatureTest	getIsMovingRight()	default	true	true
21	Creature	CreatureTest	setIsMovingRight()	snailtest.setIsMovingRight (false);	false	false
	ElementList	ElementListTest	Karena ElementList merupakan kelas template, maka dites dengan melakukan instance menggunakan salah satu instance templatennya (kelas Coin)			
22	ElementList	ElementListTest	getData()	cointest = new Coin(aqtest,vectest,10); ElementList<Coin> elemtest = new ElementList(cointest);	cointest = elemtest.get Data()	cointest = elemtest.get Data()
23	ElementList	ElementListTest	getNextTest()	elemtest.setNext(elemtest2);	elemtest2 = elemtest.get Next()	elemtest2, elemtest.get Next()
	LinkedList	LinkedListTest	Karena LinkedList merupakan kelas template, maka dites dengan melakukan instance menggunakan salah satu instance templatennya (kelas Coin)			
24	LinkedList	LinkedListTest	isEmpty()	LinkedList<Coin> test1 = new LinkedList<Coin>(); test1.isEmpty()	true	true
25	LinkedList	LinkedListTest	add()	test1.add(cointest);	cointest = test1.get(0)	cointest = test1.get(0)
26	LinkedList	LinkedListTest	getFirst()	test1.add(cointest);	cointest = test1.getFirst ().getData()	cointest = test1.getFirst ().getData()
27	LinkedList	LinkedListTest	find()	test1.add(cointest2);	test1.find(coi ntest2) = 0	test1.find(co intest2) = 0
28	LinkedList	LinkedListTest	remove()	test1.add(cointest); test1.add(cointest2); test1.remove(cointest);	test1.find(coi ntest2) = 0 test1.find(coi ntest) = -1	test1.find(co intest2) = 0 test1.find(co intest) = -1
29	LinkedList	LinkedListTest	get()	test1.add(cointest); test1.add(cointest2);	cointest2 = test1.get(1) cointest = test1.get(0)	cointest2 = test1.get(1) cointest = test1.get(0)

30	Vector2	Vector2Test	normalized()	Vector2 abnormal = new Vector2(3, 4); Vector2 normal = abnormal.normalized();	Math.atan2(abnormal.ordinate, abnormal.abscissa) = Math.atan2(normal.ordinate, normal.abscissa) normal.distance(Vector2.zero) = 1	Math.atan2(abnormal.ordinate, abnormal.abscissa) = Math.atan2(normal.ordinate, normal.abscissa) normal.distance(Vector2.zero) = 1
30	Vector2	Vector2Test	distance()	Vector2 position = Vector2.randomPosition(Vector2.zero, new Vector2(3, 4));	position.abscissa <= 3 && position.abscissa >= 0 && position.ordinate <= 4 && position.ordinate >= 0; = true	position.abscissa <= 3 && position.abscissa >= 0 && position.ordinate <= 4 && position.ordinate >= 0; = true
31	Vector2	Vector2Test	randomDirection()	Vector2 direction = Vector2.randomDirection(0, 180);	(direction.abscissa >= 0) = true direction.distance(Vector2.zero) = 1	(direction.abscissa >= 0) = true direction.distance(Vector2.zero) = 1
32	Vector2	Vector2Test	add()	Vector2 add = new Vector2(3, 4).add(new Vector2(3, 4))	add.abscissa = 6 add.ordinate = 8	add.abscissa = 6 add.ordinate = 8
33	Vector2	Vector2Test	subtract()	Vector2 subtract = new Vector2(3, 4).subtract(new Vector2(3, 4));	subtract.abscissa = 0 subtract.ordinate = 0	subtract.abscissa = 0 subtract.ordinate = 0

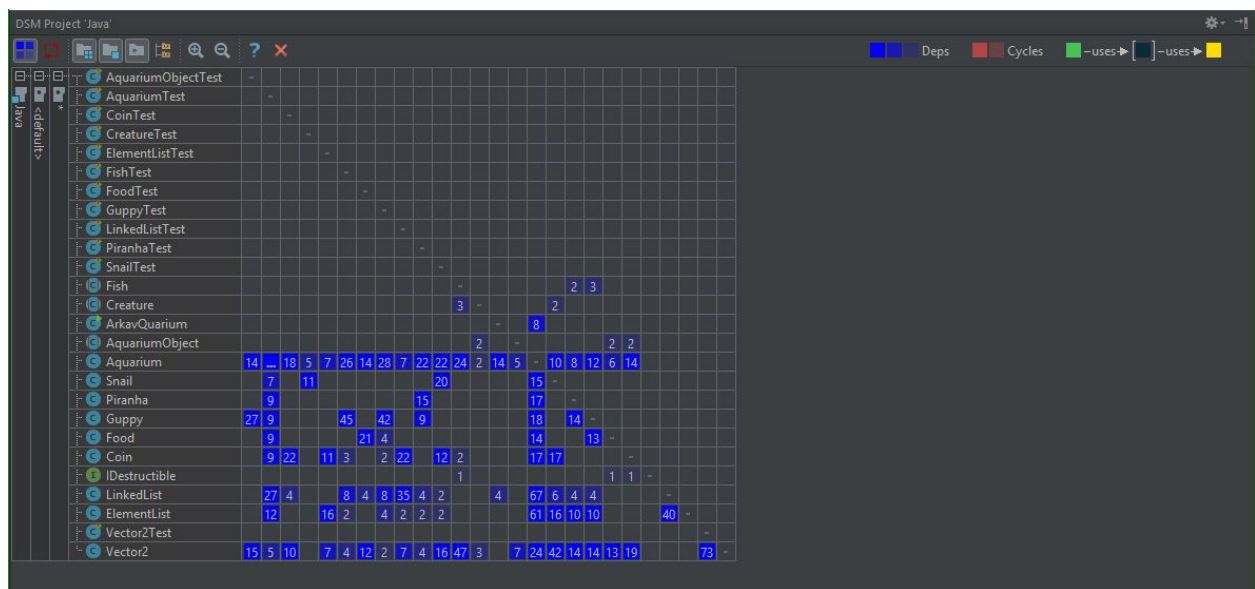
34	Vector2	Vector2Test	multiply()	Vector2 multiply = new Vector2(3, 4).multiply(5);	multiply.abscissa = 15 multiply.ordinate = 20	multiply.abscissa = 15 multiply.ordinate = 20
35	Vector2	Vector2Test	divide()	Vector2 divide = new Vector2(3, 4).divide(5);	divide.abscissa = 0.6 divide.ordinate = 0.8	divide.abscissa = 0.6 divide.ordinate = 0.8
36	Vector2	Vector2Test	toString()	Vector2(3, 4)	Vector2(3, 4).toString() = "(3, 4)"	Vector2(3, 4).toString() = "(3, 4)"
37	Vector2	Vector2Test	equals()	new Vector2(3, 4).equals(new Vector2(3, 4))	true	true
6	AquariumObject	AquariumObjectTest	getAquarium()	getAquarium();	aqtest1	aqtest1
	AquariumObject	AquariumObjectTest	getSpeed()	getSpeed();	5	5
	AquariumObject	AquariumObjectTest	getPosition()	getPosition().abscissa <= getAquarium().getSizeX(); getPosition().abscissa <= getAquarium().getSizeX();	true true	true true
	AquariumObject	AquariumObjectTest	setPosition()	setPosition(new Vector2(3.3, 4.4); getPosition().abscissa; getPosition().ordinate;	3.3 4.4	3.3 4.4
	AquariumObject	AquariumObjectTest	tick()	a = getPosition(); tick(); b = getPosition();	a != b	a != b
	Fish	FishTest	getFullTime()	getFullTime()	500	500
	Fish	FishTest	getHungryTime()	getHungryTime()	500	500
	Fish	FishTest	getMoveTime()	getMoveTime()	50	50
	Fish	FishTest	getLastMealTime()	getLastMealTime()	getAquarium().getGameTime()	getAquarium().getGameTime()
	Fish	FishTest	getLastMoveTime()	getLastMoveTime()	getAquarium().getGameTime()	getAquarium().getGameTime()
	Fish	FishTest	setLastMealTime()	setLastMealTime(1000); getLastMealTime()	1000	1000
	Fish	FishTest	setLastMoveTime()	setLastMoveTime(12000);	12000	12000

				getLastMoveTime()		
	Fish	FishTest	move()	a = getPosition(); move(); b = getPosition();	a != b	a != b
	Fish	FishTest	dropCoin()	aqtest1.add(guppy); guppy.dropCoin(3); aqtest1.getCoins().find(coin); coin.getValue();	-1 3	-1 3
	Fish	FishTest	tick()	aqtest1.add(guppy); aqtest1.getGuppies().find(guppy); setGameTime(100000000); ; guppy.tick(); aqtest1.getGuppies().find(guppy);	find(guppy) != -1 -1	find(guppy) != -1 -1
	Food	FoodTest	move()	a = getPosition(); move(); b = getPosition();	a != b	a != b
	Food	FoodTest	tick()	pos2 = new Vector2(20,60); //ordinate must be greater than aquarium.sizeY setPosition(pos2); tick(); find(food); //find food in linked list, if there's none, return -1	-1	-1
	Food	FoodTest	destruct()	add(food); //add to aquarium food.destruct(); find(food); //should return -1 if it destructed	-1	-1

	Guppy	GuppyTest	getCoinDropPeriod()	getCoinDropPeriod();	100	100
	Guppy	GuppyTest	getEatenFood()	getEatenFood();	0	0
	Guppy	GuppyTest	getLastCoinDrop()	getLastCoinDrop();	getAquarium.getTime();	getAquarium.getTime();
	Guppy	GuppyTest	getGrowthStage()	getGrowthStage(); setEatenFood(2); getGrowthStage(); setEatenFood(5); getGrowthStage();	1 2 3	1 2 3
	Guppy	GuppyTest	setEatenFood()	setEatenFood(10); getEatenFood();	10	10
	Guppy	GuppyTest	setLastCoinDrop()	setLastCoinDrop(100); getLastCoinDrop();	100	100
	Guppy	GuppyTest	eat()	add(food) add(guppy) eat()	food.getPosition()	food.getPosition()
	Guppy	GuppyTest	destruct()	add(guppy); //add to aquarium guppy.destruct(); find(guppy); //should return -1 if it destructed	-1	-1
	Guppy	GuppyTest	tick()	setGameTime(500); guppy.tick(); //should've dropped one or more coin; coin = guppy.getAquarium().getCoins().getFirst().getData(); getCoins().findCoin();	findCoin() != -1	findCoin() != -1
	Piranha	PiranhaTest	eat()	add(guppy); add(piranha); eat();	guppy.getPosition()	guppy.getPosition()
	Piranha	PiranhaTest	destruct()	add(piranha);	-1	-1

				piranha.destruct(); find(piranha); //should've return -1 if destructed from linkedList		
	Snail	SnailTest	equals()	snail1 = new Snail(aqtest1); snail2 = new Snail(aqtest2); snail1 == snail2	false	false
	Snail	SnailTest	move()	a = getPosition(); move(); b = getPosition();	a != b	a != b
	Snail	SnailTest	eat()	add(coin); add(snail); eat()	coin.getPositi on()	coin.getPosi tion()

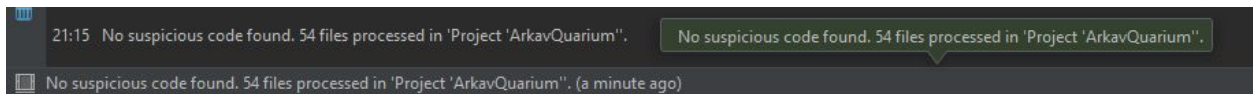
7 Pengukuran Metriks Aplikasi



Package	CC	AC	Ca(aff.)	Ce(eff.)	A	I	D	Cycle!
org.junit	0	0	1	0	0.00	0.00	1.00	-

arkavquarium	17	4	1	0	0.19	0.00	0.80	-
driver	12	0	0	2	0.00	1.00	0.00	-

8 Pengukuran Kualitas Kode Aplikasi



9 Pembagian Kerja dalam Kelompok






NIM - Nama	Kelas yang dikerjakan	Peran
13515020 - Daniel C. P. B.	Creature	Developer & Tester
	Snail	Developer & Tester
	Coin	Developer & Tester
13515116 - Aries T. S. K. A.	Fish	Developer & Tester
	Guppy	Developer & Tester
	Piranha	Developer & Tester
13516026 - Willian J. H.	Aquarium	Developer & Tester
	Aqurium Object	Developer & Tester
	Food	Developer & Tester
13516041 - Felix S. D.	LinkedList	Developer & Tester

	Vector2	Developer & Tester
--	---------	--------------------

10 Lampiran

10.1 Form Asistensi

Asistensi II

Tanggal : 18 April 2018	Catatan Asistensi: 1. Suring wajib, tapi boleh pakai yang lain juga.
Tempat : Lab Pengajaran	
Kehadiran Anggota Kelompok:	
<p>No</p> <p>NIM</p> <p>Tanda tangan</p> <p>1 13015020 </p> <p>2 13015016 </p> <p>3 13016041 </p> <p>4 13016026 </p> <p>5</p> <p>6</p>	
Tanda Tangan Asisten: 	

10.2 Log Activity Anggota Kelompok

Tanggal	13515020 Daniel	13515116 Aries	13516026 William	13516041 Felix
Rabu 18 April	Asistensi	Asistensi	Asistensi	Asistensi
Kamis 19 April	Pembagian tugas	Pembagian tugas	Pembagian tugas	Pembagian tugas
Jumat 20 April	-	-	-	-
Sabtu 21 April	-	-	-	-
Minggu 22 April	-	-	Menerjemahkan bahasa c++ ke java	Menerjemahkan bahasa c++ ke java
Senin 23 April	Menerjemahkan bahasa c++ ke java	Menerjemahkan bahasa c++ ke java	Menerjemahkan bahasa c++ ke java	Menerjemahkan bahasa c++ ke java
Selasa 24 April	Membuat JUnit	Membuat JUnit	Membuat JDocs	Menyempurnaka n Program
Rabu 25 April	Membuat JUnit, laporan	Membuat JUnit, laporan	Membuat JDocs, laporan	Menyempurnaka n Program, menyelesaikan checkStyle

10.3 Screenshot Program

