eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

Kudeaketaren eta Informazio Sistemen Informatikaren Ingeniaritzako Gradua

Programazio Modularra eta Objektuetara Bideratutako Orientazioa — 2. lauhilabetea

The Ol' Western

Jon Ander Asua, Adei Arias, Ander Prieto eta Iker Monzón — 31. taldea

AURKIBIDEA

1.- Sarrera

1.1.- Jokoaren deskribapena

Guk, **while(furros){uwu}** taldeak, ***The Ol' Western*** izeneko jokoa sortu dugu. Istorio lineal baten ezaugarriak dituen jokoa da, baita logika (asmakizunak baititu) eta abentura.

Western batean girotuta dagoen jokoa da; jokalaria pistolari bat izango da, Nevadako Ryolite herrira helduko dena, gaizkileen banda baten atzetik. Helburua banda harrapatzea eta kide guztiak erailtzea izango da, baina banda harrapatzeko hainbat leku edo egoeratik igaro beharko da.

Jokoa hiru egoeratan banatuta dago: Saloia, hilerria eta banketxea. Lehenengoan tabernaria, prostituta eta gizon zaharra daude, eta hirurek emandako informazioarekin izkutatuta dagoen kutxagogor bat ireki behar da. Hori egitean, gutun bat agertuko da non hilerri bat aipatzen den, eta zuzenean hurrengo egoerara, hilerrira, pasatuko da. Hemen ehorzlea eta apaiza soilik daude, eta elizara sartzea dugu helburu. Hori lortzean, gaizkile zauritu bat aurkituko dugu, banda harrapatzeko banketxera joateko esango diguna. Elkarrizketa honen ondoren, banketxera helduko gara.

Banketxean banda aurkituko dugu, zazpi gaizkilez osatuta. Helburua, arestian aipatuta dagoenez, etsai guztiak erailtzea da. Hemen bi amaiera posible daude:

1. Etsaiak erailtzea: Hau gertatzean, istorioari amaiera emateko testu bat agertuko da, baita zorionak emateko mezua. Ondoren, jokoa amaituko da.
2. Protagonista bizitzarik gabe gelditzea: Hau gertatzean, jokalariari jokatzen jarraitu nahi badu eskatuko zaio
    a. Baiezko erantzunean: Banketxea egoera berriz hasiko da. Etsaiak berriz agertuko dira eta protagonistaren bizitza topera igoko da berriro
    b. Edozein beste erantzunean: "Game over" testua agertuko da eta jokoa bukatuko da.

Jokatzeko modua hurrengokoa izango da: Posibleak diren ekintzen zerrenda pantailaratuko da, zenbakizko lista eran, eta burutu nahiko den ekintzaren zenbakia kokatuz nahikoa izango da. Hasierako bi egoeretan, posiblea den ekintza nagusia elkarrizketa da. Hala ere, NPC-ek (beste pertsonaiek) ez dute beti informaziorik emango, ausazkoa da. Beraz, baliteke pertsonaia batekin behin baino gehiagotan hitz egin behar egitea beharrezko informazioa lortzeko (nabarikoa da noiz gertatuko den).

Bukatzeko, esan beharra dago egoera bakoitzak bere mapa duela, eta ekintza bakoitza burutzerakoan mapa eguneratu eta berriz inprimatuko da.

1.2.- Proiektuaren helburuak

Bitan banatu ditzakegu helburuak: Lehenengo mailakoak eta bigarren mailakoak.

Argi dago zeintzuk diren lehenengo mailako helburuak. Hauen artean, jokoa hasieratzea dago, esaterako; hau da, behar diren akzioen lista sortzea, pertsonaia hasieratzea eta matrizea inprimatu, baita ere banketxe egoeran ditugun tresnekin tiroketa sistema on bat inplementatzea.

Bigarren mailako helburu nagusiena matrizea ondo kudeatzea da, hau da, egoera guztiak kontsolan ondo inplementatzea eta handik jokalaria mugitzea. Beste aldetik kodea optimizatzea, proiektua amaitu ondoren hainbat metodo eta kode lerro ezabatu genituen.

2.- Plangintza eta kudeaketa

Klase diagraman ikusiko den bezala (3.1.1 atalean), bi adarkatze nagusi ditu, lehenengoa Pertsonaia klasearekin erlazioa dutenak, eta bigarrena Egoera klasearekin erlazioa duena. Horregatik, klase diagrama egin ostean proiektuaren kodea egiteko bi azpitalde sortu genituen, talde bakoitzak adarkatze batekin:

- Adei eta Ander: ListaPertsonaiak eta harekin dauden erlazioaz okupatu ziren, hau da, jokoan agertzen diren pertsonaien kudeaketaz okupatu egin ziren.
- Iker eta Jon Ander: ListaEgoerak eta klase horrekin erlazionatutako klaseak osatu zuten, hau da, dauden egoerak kudeatzeko balio duten klaseak. Adei eta Anderrek adarkatze honetan ere garrantzia izan dute.

Dauden beste klase batzuk (FitxategiakIrakurri, Dadoa eta Teklatua) eGelatik edo internetetik atera genituen, baina Teklatua klasean pare bat metodo guk sortu genituen, adibidez emanEnter metodoa (Klase diagramaren atalean azalduko dena).

Beste aldetik, proiektuaren diseinua, klase eta sekuentzia diagrama Jon Anderrek egin du baina Adei eta Anderren laguntza askorekin. Atal hau aldaketa asko izanagatik ez da egon problema askorik aldatzerako orduan.

PHDa eta Memoria idazteaz denek okupatu egin gara, bakoitzak ahal izan duen moduan gauzak gehitzeaz okupatu egin da. Googlek eskainitako Drive plataformaren bidez egin dugu.

Azkenik, JUnitak inplementatu ditugu. Metodo askok ez dute JUnitik behar izan, eta beste batzuentzat ezinezkoa da hau egitea; izan ere, beste metodo edo atributuen menpe daude.

	ORDUAK	ANDER	JON ANDER	ADEI	IKER
KLASE DIAGRAMA	**8**	4	2	2	–
SEKUENTZIA DIAGRAMA	**10**	–	8	2	–
KODEA	**40**	18	8	14	–
JUNITAK	**5**	–	2	2	1
SALBUESPENAK	**2**	–	1	1	–
PHD	**4**	1	1	1	1
AURKEZPENA	**5**	1.5	2	1.5	–
MEMORIA	**15**	5	5	5	–

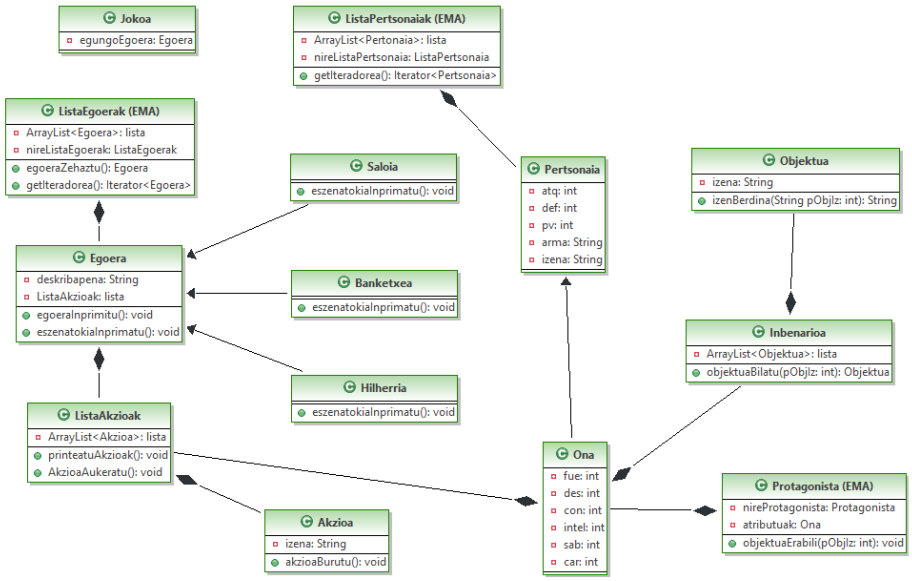
3.- Diseinua

3.1.- Klase diagrama

3.1.1.- Klase diagramaren eboluzio edo garapena

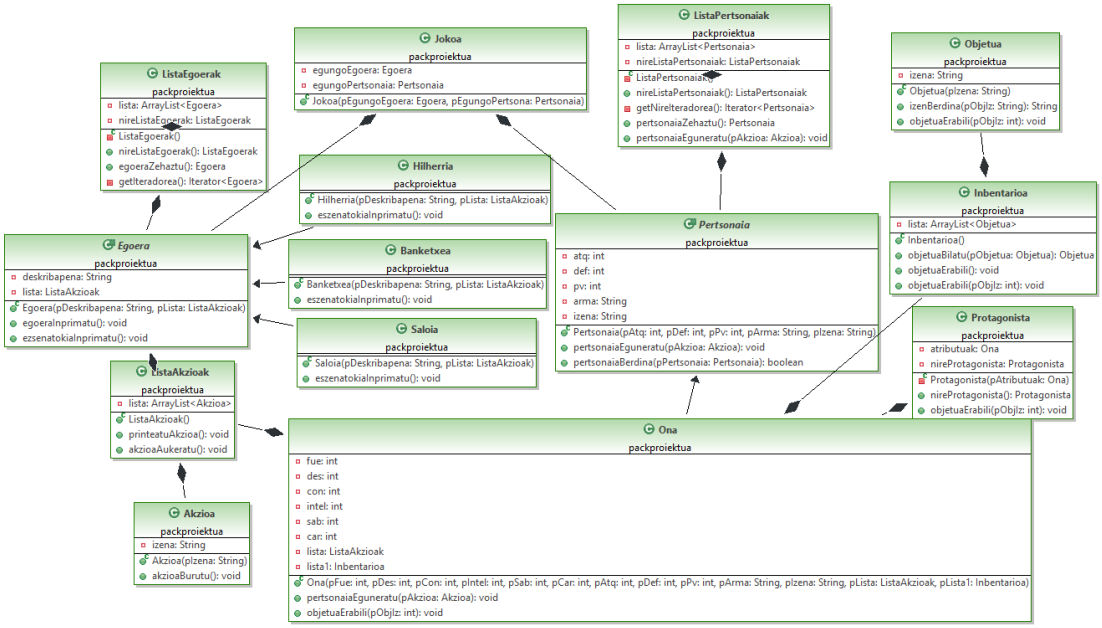
Proiektua aurrera joan den ahala, gure diseinuan aldaketak egon dira. Hau da, klase diagramak aldaketak jaso ditu etengabe. Hau izan da bere garapena:

- Martxoak 17:



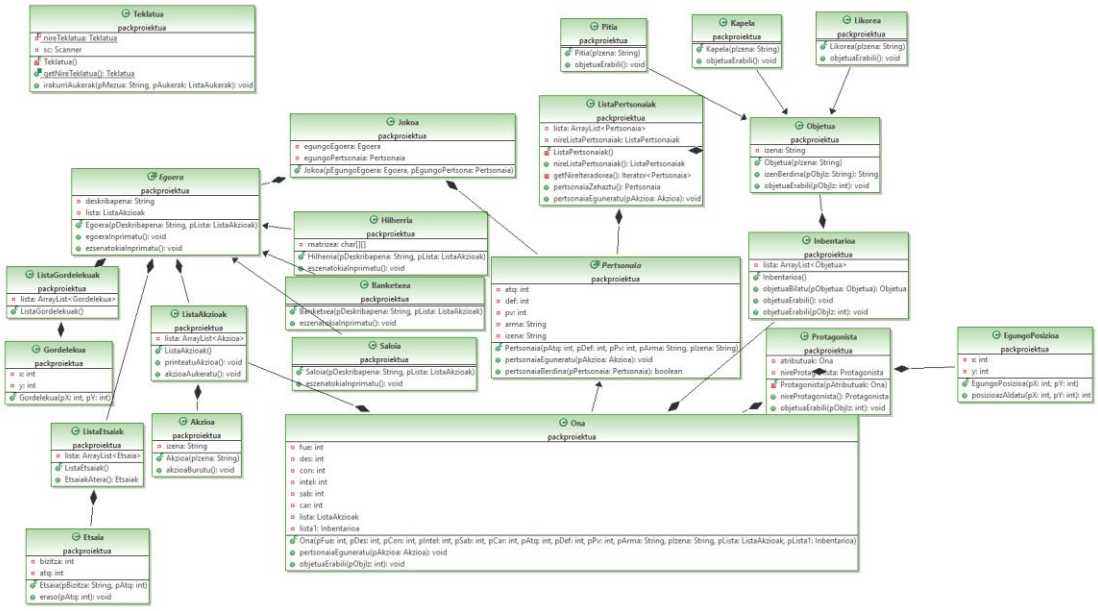
PHDa egiterakoan, hau izan zen ideia nagusia. Amaierako klase diagramaren bizkarrezurra da.

- Martxoak 31:



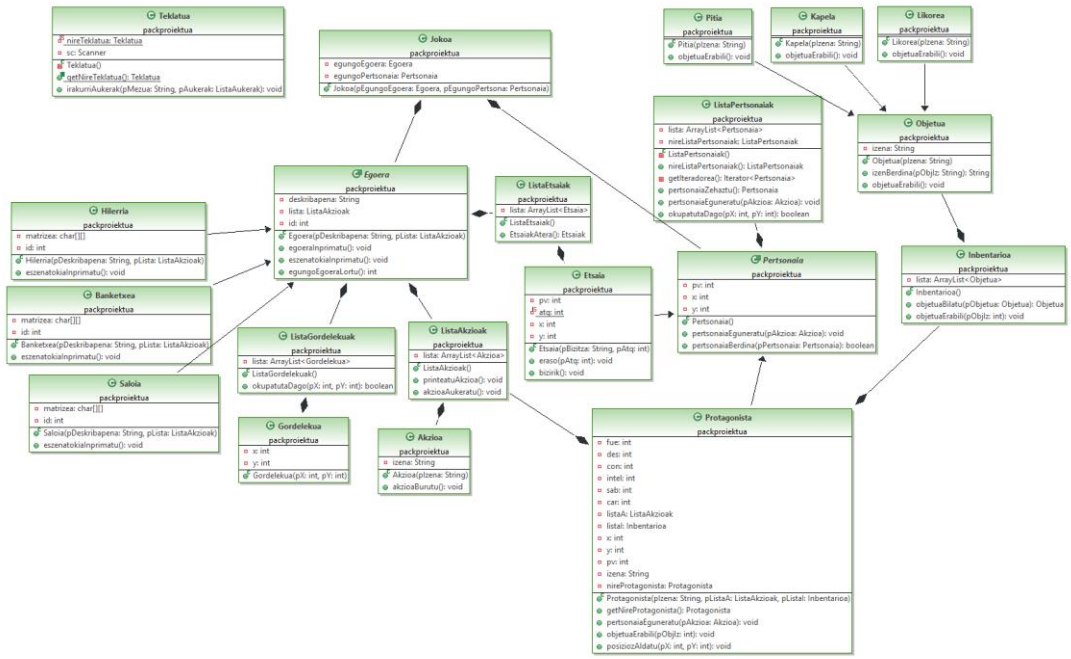
Jokoa klasean main metodoa egongo zela, Egoera eta Pertsonaia klaseak (bi adar nagusiak) erlazioa izango zuten honekin

- Apirilak 17:



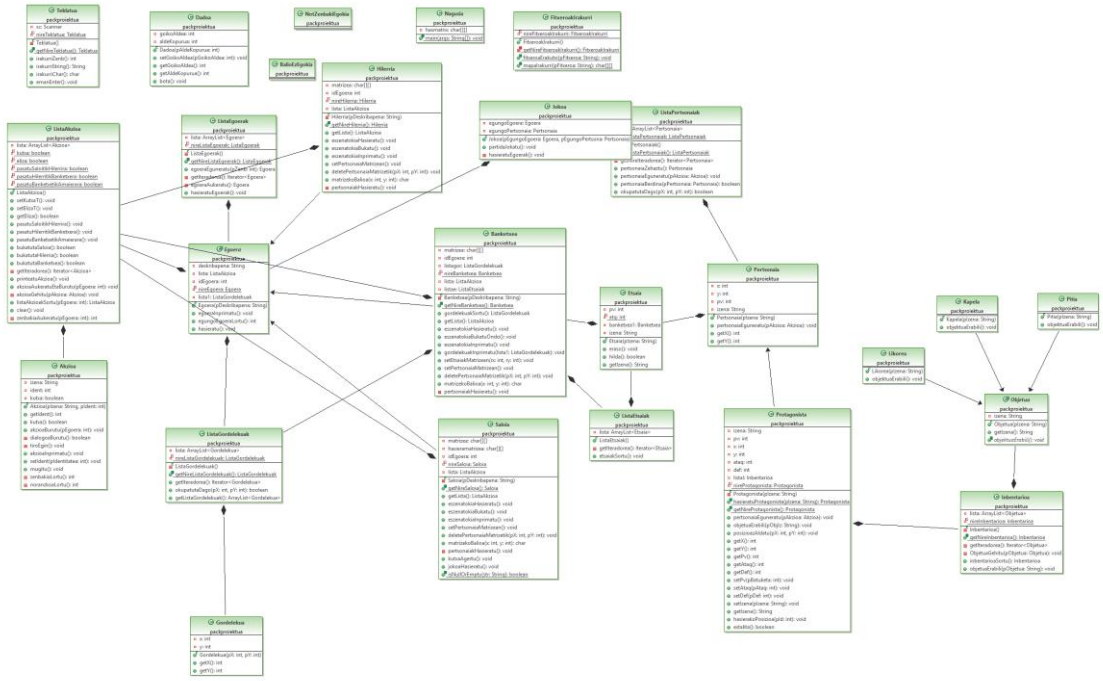
Inbentarioko objektuak definitu genituen. Gainera, Gordelekua eta Etsaia klaseak eta hauen listak ere kokatu genituen.

❖ Apirilak 20:



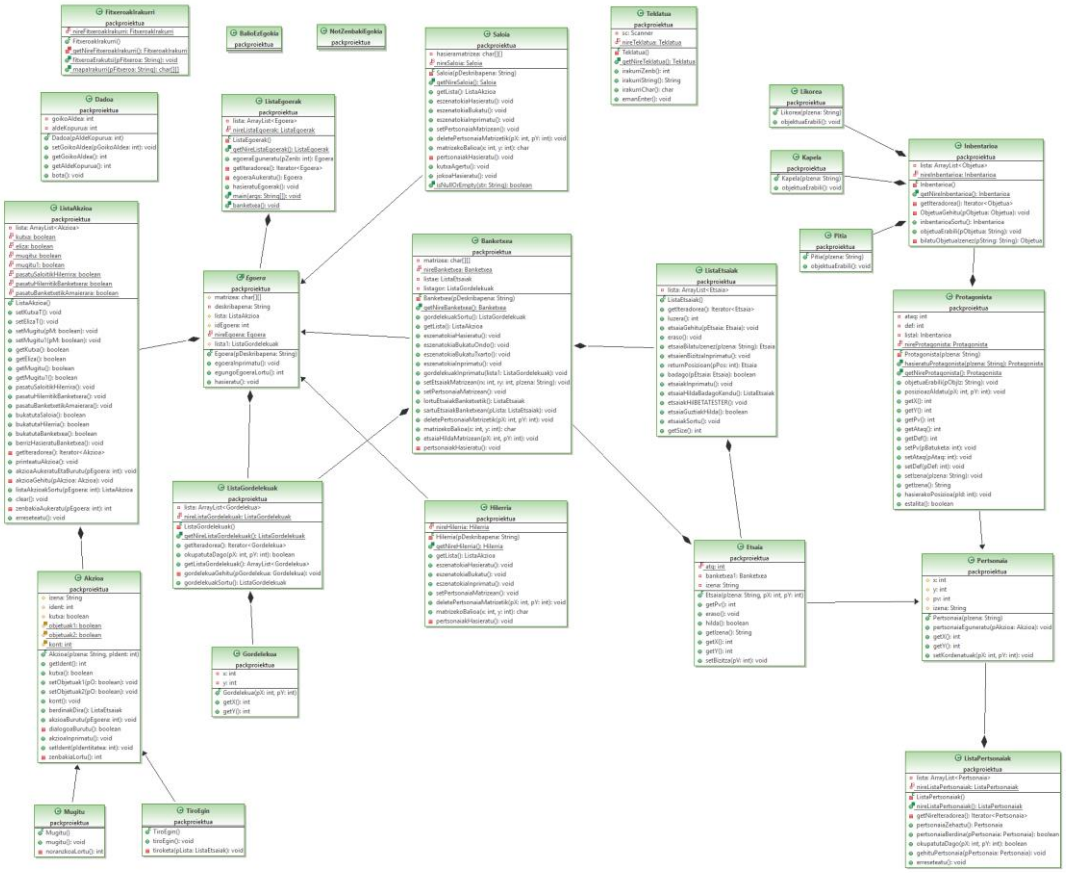
Ona klasea kendu genuen (horrela protagonista bakarrik ibiliko zen) eta Etsaia Pertsonaiaren herentzia izatea ere erabaki genuen

❖ Maiatzak 4:



Aurkezpenerako eguneratu genuen hau. Metodoak jarri, ordenatu eta gordelekuak Banketxea klasearekin soilik lotu genituen, eta berdina etsaiekin. Teklatua, dadoa, salbuespenak… klaseak ere kokatu genituen.

❖ Ekainak 6:



Klase diagrama finala da. Aurkezpenean jasotako aholkuak erabiliz Nagusia klasea ListaEgoerak EMAn sartu genuen eta horrekin main metodoa ere. Herentzia sortu genuen Akzioa klasetik eta protected egoera behar zuten atributuen pribatasuna eguneratu genuen ere. Klase diagrama hau A3 tamainako orri baten inprimatuta egongo da ere.

3.1.2.- Klase bakoitzeko metodo eta atributuen azalpena

- Metodo bat *kurtsiban (italikan)* egoteak metodo pribatua dela adierazten du.
- Atributuetan paresentesien barnean dagoena atributu horren mota adierazten du.

ListaEgoerak (EMA):

- Atributuak:
  - lista: Akzioaz osatutako lista bat da.
  - nireListaEgoerak: Atributu hau erabiliko dugu Singleton patroia egiteko.
- Metodoak:

- ListaAkzioa(): ListaAkzioa klasearen eraikitzailea.
- getNireListaEgoerak(): Singleton patroiaren bidez ListaEgoerak bakarra bueltatzen dizu.
- hasieratuEgoerak(): Dauden hiru egoerak hasieratzen dira, egoera bakoitzean berriro sartzean hasieratuta egoteko.
- main(args: String[]): Metodo hau jokoaren partida bat jokatzeko balio du.
- banketxea(): Banketxea egoera martxan jartzen du, eta bertan dauden eta egin ahal diren gauza guztiak ere.

Saloia:

- Atributuak:
  - Klase hau Egoera klasean protected dauden atributuak heredatzen ditu, honetaz aparte bi atributu gehiago ditu.
  - hasieramatrizea(char [][]): Egoera honen mapa inprimatzeko erabiltzen da.
  - nireSaloia(Saloia): Singleton patroia egiteko balio duen atributu estatikoa.
- Metodoak:
  - Saloia(pDeskribapena: String): Klasearen eraikitzaile pribatua.
  - getNireSaloia(): Saloia bakarra bueltatzen duen metodo estatikoa, singleton patroiaren bitartez.
  - getLista(): Egoera horren ListaAkzioak bueltatzen dizu.
  - eszenatokiaHasieratu(): Saloia Egoeraren eszenatokia hasieratzen du, hau da, interazioak hasi baino lehen dagoen Saloia inprimitzen du.
  - eszenatokiaBukatu(): Jarritako helburua betetzen denean “Saloia/Saloia_Bukatuta.txt” fitxeroa inprimatzen du.
  - eszenatokiaInprimatu(): Matrizean dauden elementuak inprimatzen ditu.
  - setPertsonaiaMatrizean(): Pertsonaia matrizean jartzen du.
  - deletePertsonaiaMatrizetik(pX,pY:int): Pertsonaia dagoen posiziotik (pX eta pY kordenatuetan) kentzen du.
  - matrizekoBalioa(pX,pY:int): pX eta pY posizioetan dagoen elementua ezabatzen du.
  - *pertsonaiakHasieratu():* Egoeran dauden pertsonaiak hasieratzen ditu eta defektuz dauden lekuetan jartzen ditu.
  - kutxaAgertu(): Egoera honetan dagoen ListaAkzioak atributuan kutxara joateko eskubidea ematen digu.

- jokoaHasieratu(): Jokoaren hasierapena egiten du, hau da hasiera, aurkezpena eta azalpena ematen duten fitxategiak erakusten ditu, baita protagonistaren izena eskatzen du.
- isNullOrEmpty(str:String): Sartutako Stringa hutsa den ala ez esaten dizu, hutsa bada True bat bueltatuko dizu eta False null ez bada.

Hilerria:

- Atributuak:
  - Klase hau Egoera klasean protected dauden atributuakheredatzen dtu, honetaz aparte bi atributu gehiago ditu.
  - nireHilerria(Hilerria): Singletton patroia egiteko balio duen atributu estatikoa.
- Metodoak:
  - *Hilerria(pDeskribapena: String):* Klase honen eraikitzaile pribatua da.
  - getNireHilerria(): Hilerri bakarra bueltatzen duen metodo estatikoa, singleton patroiaren bitartez.
  - getLista(): Egoera horren ListaAkzioak bueltatzen dizu.
  - eszenatokiaHasieratu(): Hilerria Egoeraren eszenatokia hasieratzen du, hau da, interazioak hasi baino lehen dagoen Saloia inprimitzen du.
  - eszenatokiaBukatu(): Jarritako helburua betetzen denean "Saloia/Saloia_Bukatuta.txt" fitxeroa inprimatzen du.
  - eszenatokiaInprimatu(): Matrizean dauden elementuak inprimatzen ditu.
  - setPertsonaiaMatrizean(): Pertsonaia matrizean jartzen du.
  - deletePertsonaiaMatrizetik(pX,pY:int): Pertsonaia dagoen posiziotik (pX eta pY kordenatuetan) kentzen du.
  - matrizekoBalioa(pX,pY:int): pX eta pY posizioetan dagoen elementua ezabatzen du.
  - *pertsonaiakHasieratu():* Egoeran dauden pertsonaiak hasieratzen ditu eta defektuz dauden lekuetan jartzen ditu.

Banketxea:

- Atributuak:
  - Klase hau Egoera klasean protected dauden atributuakheredatzen dtu, honetaz aparte bi atributu gehiago ditu.
  - matrizea(char [][]): Egoera honen mapa inprimatzeko erabiltzen da.
  - nireBanketxea(Banketxea): Singleton patroia egiteko balio duen atributu estatikoa.

    - listaE (ListaEtsaiak): Egoera honen ListaEtsaiak gordetzeko balio du.
    - listaGor (ListaGordelekuak): Egoera honen ListaGordelekuak gordetzeko balio du.
- Metodoak:
    - *Banketxea(pDeskribapena: String):* Klase honen eraikitzaile pribatua da.
    - getNireBanketxea(): Banketxe bakarra bueltatzen duen metodo estatikoa, singleton patroiaren bitartez.
    - getLista(): Egoera horren ListaAkzioak bueltatzen dizu.
    - eszenatokiaHasieratu(): Banketxea Egoeraren eszenatokia hasieratzen du, hau da, interazioak hasi baino lehen dagoen Saloia inprimitzen du.
    - eszenatokiaBukatu(): Jarritako helburua betetzen denean "Saloia/Saloia_Bukatuta.txt" fitxeroa imprimatzen du.
    - eszenatokiaInprimatu(): Matrizean dauden elementuak inprimatzen ditu.
    - setPertsonaiaMatrizean(): Pertsonaia matrizean jartzen du.
    - deletePertsonaiaMatrizetik(pX,pY:int): Pertsonaia dagoen posiziotik (pX eta pY kordenatuetan) kentzen du.
    - matrizekoBalioa(pX,pY:int): pX eta pY posizioetan dagoen elementua ezabatzen du.
    - gordelekuakSortu(): listaG atributua bueltatzen du.
    - gordelekuakInprimatu(lista1:ListaGordelekuak): ListaGordelekuak eszenatokian jartzen ditu, hau da, inprimatzen ditu.
    - setEtsaiakMatrizean(pX,pY:int,pIzena:String): Etsaia eszenatokian dagokion lekuan inprimatzen du.
    - lortuEtsaiakMatrizetik(): listaE atributua bueltatzen du.
    - etsaiaHildaMatrizean(pX,pY:int):Etsai bat hiltzerakoan, etsai hori dagoen posizioan 'X' bat jartzen du.
    - *pertsonaiakHasieratu():* Egoeran dauden pertsonaiak hasieratzen ditu eta defektuz dauden lekuetan jartzen ditu.

ListaAkzioak:

- Atributuak:
    - lista(ArrayList<Akzioa>): Akzioaz osatutako ArrayList bat.
    - kutxa(boolean): Kutxa irekitzeko aukera emateko balio du.
    - eliza(boolean): Elizan sartzeko aukera emateko balio du.
    - mugitu(boolean): Pertsonaia mugitzen ari den adierazten du.

- mugitu1(boolean): Pertsonaia mugitzen ari den adierazten du (beste metodo batetan erabiltzen da).
- pasatuSaloitikHilerrira(boolean): Jokalaria Saloian jarritako helburua bete duen jakiteko.
- pasatuHilerritikBanketxera(boolean): Jokalaria Hilerrian jarritako helburua bete duen jakiteko.
- pasatuBanketxetikAmaierara(boolean): Jokalaria Banketxean jarritako helburua bete duen jakiteko.

- Metodoak:
  - ListaAkzioa(): ListaAkzioa klasearen eraikitzailea.
  - setKutxaT(): kutxa atributua "True"ra jartzen du.
  - setElizaT(): eliza atributua "True"ra jartzen du.
  - setMugitu(pM:boolean): mugitu atributua "True"ra jartzen du.
  - setMugitu1(pM:boolean): mugitu1 atributua "True"ra jartzen du.
  - getKutxa(): kutxa atributua bueltaten dizu.
  - getEliza(): eliza atributua bueltaten dizu.
  - getMugitu(): mugitu atributua bueltaten dizu.
  - getMugitu1(): mugitu1 atributua bueltaten dizu.
  - pasatuSaloitikHilerrira(): pasatuSaloitikHilerrira truera jartzen du.
  - pasatuHilerritikBanketxera(): pasatuHilerritikBanketxera truera jartzen du.
  - pasatuBanketxetikAmaierara(): pasatuBanketxetikAmaierara truera jartzen du.
  - bukatuSaloia(): pasatuSaloitikHilerrira bueltatzen du.
  - bukatuHilerria(): pasatuHilerritikBanketxera bueltatzen du.
  - bukatuBanketxea(): pasatuBanketxetikAmaierara bueltatzen du.
  - berrizHasieratuBanketxea(): Protagonistaren PV=0 denean Banketxearen egoera berriro hasteko aukera ematen du, hau da, pasatuBanketxetikAmaierara atributua falsera jartzen du.
  - *getIteradorea():* Akzioaz osatutako ArrayListaren Iteratora bueltatzen du.
  - printeatuAkzioa(): Akzioa inprimitzen du.
  - akzioaAukeratuEtaBurutu(): Listaren barnean Akzio bat aukeratzeko eta akzio hori burutzeko.
  - *akzioaGehitu(pAkzioa:Akzioa):* Listara akzio berri bat gehitzeko.
  - listaAkzioaSortu(pEgoera:int): Egoeraren arabera ListaAkzio bat edo beste ezberdin bat bueltatzen du.
  - clear(): ListaAkzioa erreseteatzen du.

- *zenbakiaAukeratu(pEgoera:int):* Akzioa aukeratzeko orduan zenbakia irakurtzen eta egokia den ala ez esaten duen metodoa.

Akzioa:

- Atributuak:
  - Klase honen atributu GUZTIAK protected dira.
  - izena (String): Akzioaren izena gordetzen du.
  - ident (int): Akzioaren identifikadore numeriko bat da.
  - kutxa (boolean): ListaAkzioaren atributu berdina da.
  - objetuak1 (boolean): Pitia soilik behin erabiltzea baimentzen du.
  - objetuak2 (boolean): Kapela soilik behin erabiltzea baimentzen du.
  - kont (int): Likorea soilik birritan erabiltzea baimentzen du.
- Metodoak:
  - Akzioa(pIzena: String, pIdent: int): Akzioa klasearen eraikitzaile publikoa da.
  - getIdent(): ident atributua bueltatzen du.
  - getKutxa(): kutxa atributua bueltatzen du.
  - setObjetuak(pO: boolean): objetuak1 “True” baliora kokatzen du
  - setObjetuak2(pO: boolean): objetuak2 “True” baliora kokatzen du
  - kont(): kontadorea batean inkremetatzen du.
  - berdinakDira(): Etsaiaren eta zure koordenatuak berdinak diren konprobatzen du. Hala bada, etsaia ListaEtsaiak baten sartzen du
  - akzioaBurutu(): Ea akzioa zein den gauza bat edo beste bat egiten du, hau da, akzioa burutzen du.
  - dialogoaBurutu(): Boolean bat bueltatzen du adierazten ea dialogoa egokia den ala ez, hau random batekin kalkulatzen da.
  - akzioaInprimatu(): Akzioa inprimatzen du.
  - setIdent(pIdent:int): pIdent Akzioaren ident atributua bihurtzen du.
  - *zenbakiaLortu():* Teklatua irakurtzen du eta sartutako zenbakia egokia den ala ez adierazten du, egokia bada zenbaki hori buelltatzen du.

Mugitu:

- Atributuak:
  - Akzioa ama klasearen atributuak heredatzen ditu.
- Metodoak:
  - Mugitu(): Mugitu klasearen eraikitzailea.
  - mugitu(): Protagonistaren kordenatuak aldatzen ditu, hau da, pertsonaia miugitzen du.

- *noranzkoaLortu():* Teklatutik karaktere bat irakurtzen du eta egokia den ala ez begiraten du. Karakterea egokia bada integer bat bueltatzen du.

TiroEgin:

- Atributuak:
  - Mugitu klasea bezala, atributuak bere ama klasetik hartzen ditu.
- Metodoak:
  - TiroEgin(): Tiro egin klasearen eraikitzailea.
  - tiroEgin(): Tiro egitearen akzioa burutzen du, hau da, zuk aukeratutako etsaiaren bizitza dekrementatzen du.
  - tiroketa(ListaEtsaiak pLista): Zure eskura dauden etsaien artean aukeratzeko balio duen metodoa.

ListaGordelekuak (EMA):

- Atributuak:
  - lista(ArrayList<Gordelekua>): Gordelekuz osatutako ArrayList bat .
  - nireListaGordelekuak(ListaGordelekuak): Atributu hau erabiltzen da Singleton patroia inplementatzeko.
- Metodoak:
  - *ListaGordelekuak():* Klase honen eraikitzaile pribatua.
  - getNireListaGordelekuak(): "ListaGordelekuak" bakarra bueltatzen du, Singleton patroiaren bidez.
  - geIteradorea(): "lista"ren iteratora bueltatzen dizu.
  - okupatutaDago(pX,pY:int): pX eta pYk adierazitako kordenatuak okupatuta dauden ala ez esaten dizu.
  - *gordelekuaGehitu(pGordelekua: Gordelekua)*: Gordeleku berri bat "lista" atributuaren barnean gordetzen du.
  - gordelekuakSortu(): Jadanik predefinituta ditugun gordelekuaz osatutako zerrenda bueltatzen du.

Gordelekuak:

- Atributuak:
  - x (int): Gordelekuaren x kordenatua.
  - y (int): Gordelekuaren y kordenatua.
- Metodoak:
  - Gordelekua(pX,pY:int): Klase honen eraikitzaile publikoa.
  - getX(): "x" atributua bueltatzen du.
  - getY(): "y" atributua bueltatzen du.

ListaPertsonaiak:

- Atributuak:
  - lista: Pertsonaiaz osatutako zerrenda bat.
  - nireListaPertsonaiak (ListaPertsonaiak): Singleton patroia egiteko balio duen atributu estatikoa.
- Metodoak:
  - *ListaPertsonaiak():* Klase hone eraikitzaile pribatua.
  - nireListaPertsonaiak(): Singleton patroia eginez ListaPertsonaiak bakarra bueltatzen du.
  - *getIteradorea():* Pertsonaiaz osatutako zerrendaren iteradorea bueltatzen du.
  - pertsonaiaBerdina(pPertsonaia: Pertsonaia): Sartutako pertsonaia zerrendaren barnean dagoen ala ez adierazten du.
  - okupatutaDago(pX,pY:int): Sartutako kordenatuetan Pertsonaia bat dagoen ala ez adierazten du.
  - gehituPertsonaia(): Pertsonaia bat zerrendan sartzen du.
  - erreseteatu(): Zerrenda husten du.

Pertsonaia:

- Atributuak:
  - Dituen atributu guztiak protected dira, haren seme klaseak heredatu ahal izateko.
  - x(int): Pertsonaiaren x kordenatua.
  - y(int): Pertsonaiaren y kordenatua.
  - pv(int): Pertsonaiaren bizitza kantitatea.
  - izena(String): Pertsonaiari jarritako izena.
- Metodoak:
  - Pertsonaia(pIzena: String): Pertsonaiaren erakitzailea.
  - pertsonaiaEguneratu(pAkzioa:Akzioa): Akzioa burutzeko behar diren pertsonaiaren egunerapenak egiten ditu, adibidez tiro bat jasotzen badu, pertsonaiaren bizitza (pv) jaitziko da.
  - getX(): "x" atributua bueltatzen du.
  - getY(): "y" atributua bueltatzen du.

Etsaia:

- Atributuak:
  - Pertsonaia ama klasearen seme bat denez haren atributuak heredatzen ditu.
  - atq(int): Indarra.

- Metodoak:
    - Etsaia(pIzena: String, pX,pY:int): Klase honen eraikitzailea.
    - getPv(): “pv” atributua bueltatzen du.
    - eraso(): Protagonistaren aurka erasotzearen akzioa egiten du.
    - hilda(): Etsaiak bizitza ez duela adierazten du, hau da haren pv<=0.
    - getIzena(): “izena” atributua bueltatzen du.
    - getX(): “x” aldagaia bueltatzen du.
    - getY(): “y” aldagaia bueltatzen du.
    - setBizitza(pV:int): pv=pV egiten da, hau da, bizitza berri bezala sartutako parametroa jartzen da.

ListaEtsaiak:

- Atributuak:
    - lista (ArrayList<Etsaia>): Etsaiaz osatutako zerrenda.
- Metodoak:
    - ListaEtsaiak(): Klasearen eraikitzailea.
    - getIteradorea(): Zerrendaren iteradorea bueltatzen du.
    - luzera(): zerrendaren luzera bueltatzen du.
    - etsaiaGehitu(pEtsaia: Etsaia): Etsaia bat zerrendara gehitzen du.
    - eraso(): Etsai bakoitza eraso egiten du.
    - etsaiaBulatuIzenez(pIzena: String): String bat sartuta kointziditzen duen etsaia bueltatzen du.
    - etsaienBizitzaInprimatu(): Etsai bakoitzaren Bizitza puntuak (pv) bueltatzen du.
    - returnPosizioan(pPos): pPos posizioan dagoen etsaia bueltatzen du.
    - badago(pEtsaia: Etsaia): Etsai bat zerrendan dagoen ala ez adierazten du.
    - etsaiakInprimatu(): Etsaiak eszenatokian inprimatzen ditu.
    - etsaiaHildaBadagoKendu(): Hilda dauden Etsaiak zerrendatik kentzen ditu.
    - etsaiaGuztaikHilda(): Etsai guztiak hilda dauden ala ez adierazten du.
    - etsaiakSortu(): Etsaiak “Banketxea” egoeran sortzen ditu.

Protagonista (EMA):

- Atributuak:
    - Pertsonaiaren atributuak heredatzen ditu.
    - ataq (int): Protagonistaren indarra.
    - def (int): Potagonistaren defentsa.

    - listaL (Inbentarioa): Proganistaren Inbentarioa.
    - nireProtagonista: Singleton patroia egiteko balio duen atributu estatikoa.
- Metodoak:
    - *Protagonista(pIzena: String):* Protagonistaren eraikitzailea.
    - hasieratuProtagonista(pIzena: String): Protagonistari hasierako balioak jartzen dion metodoa.
    - getNireProtagonista(): Singleton patroian oinarrituta Protagonista bakarra bueltatzen du.
    - objetuaErabili(pObj: String): Objetu bat erabiltzen du, hau protagonistaren atributuen balioak aldatzen ditu.
    - posizioazAldatu(pX,pY: int): "x" atributuaren balioa pX izango da eta "y" atributuarena pY.
    - getX(): "x" atributua bueltatzen du.
    - getY(): "y" atributua bueltatzen du.
    - setPv(pBatuketa: int): Sartutako balioa Protagonistaren "pv" atributuaren balioa izatera pasatuko da.
    - setAtaq(pAtaq:int): Sartutako balioa Protagonistaren "atq" atributuaren balio berria izatera pasatuko da.
    - setDef(pDef: int): Sartutako balioa Protagonistaren "def" atributuaren balio berrira izatera pasatuko da.
    - setIzena(pIzena: String): Protagonistaren "izena" atributuaren balioa aldatuo da, balio berria pIzena izango da.
    - getIzena(): "izena" atributua bueltatzen du.
    - hasierakoPosizioa(pId: int): Ea zein egoeran dagoen Protagonistari x eta y kordenatu ezberdinak esleituko zaizkio.
    - estalita(): Protagonista ea gordeleku batean estalita dagoen ala ez adierazten du.

Inbentarioa (EMA):

- Atributuak:
    - lista (ArrayList <Objetua>): Objetuz osatutako zerrenda bat.
    - nireInbentarioa(Inbentarioa): Singleton patroia egiteko balio duen atributu estatikoa.
- Metodoak:
    - *Inbentarioa():* Klasearen eraikitzailea.
    - getNireInbentarioa(): Singleton patroia jarraituz Inbentario bakarra bueltatzen du.
    - *getIteradorea():* Zerrendaren iteradorea bueltatzen du.

- *objetuaGehitu(pObjetua: Objetua):* Zerrendara sartutako objetua gehitzen du.
- inbentarioaSortu(): Inbentarioa sortzen du.
- objetuaErabili(): Objetu bat erabiltzen du, hau protagonistaren atributuen balioak aldatzen ditu.
- *bilatuObjetuaIzenez(pString: String):* Izenaren bidez objetu bat bilatzen eta bueltatzen du.

Objetua:

- Atributuak:
  - izena(String): Objetuaren izena.
- Metodoak:
  - Objetua(pIzena:String): Klasearen eraikitzailea.
  - getIzena(): "izena" atributua bueltatzen du.
  - objetuaErabili(): *Metodo abstraktoa*. Objetu bat erabiltzen du, hau protagonistaren atributuen balioak aldatzen ditu.

Pitia:

- Atributuak:
  - Objetua ama klasetik heredatzen ditu.
- Metodoak:
  - Pitia(): Klasearen eraikitzaikea.
  - objetuaErabili(): pv+10 egiten du.

Kapela:

- Atributuak:
  - Objetua ama klasetik heredatzen ditu.
- Metodoak:
  - Kapela(): Klasearen eraikitzailea.
  - objetuaErabili(): pv+50 egiten du.

Likorea:

- Atributuak:
  - Objetua ama klasetik heredatzen ditu.
- Metodoak:
  - Likorea(): Klasearen eraikitzailea.
  - objetuaErabili(): pv+100 egiten du.

Dadoa:

- Atributuak:

    - goikoAldea(int): Dadoaren goiko aldea.
    - aldeKop(int): Dadoaren alde kopurua.
- Metodoak:
    - Dadoa(pAldeKop:int): Klasearen eraikitzailea.
    - setGoikoAldea(pGoikoAldea:int): goikoAldearen balio berria pGoikoAldea izango da.
    - getGoikoAldea(): "goikoAldea" bueltatzen du.
    - getAldeKopurua():"aldeKop" bueltaten du.
    - bota(): Dadoa botatzen da, ondorioz "goikoAldea" atributua balioz aldatzen da, baina inoiz e aldeKop baino handiagoa edo txikiagoa den zenbaki batekin.

Teklatua (EMA):

- Atributuak:
    - sc(Scanner): Eskanerra.
    - nireTeklatua(Teklatua): Singleton patroia aplikatzeko balio duen atributu estatikoa.
- Metodoak:
    - *Teklatua():* Klasearen eraikitzailea.
    - getNireTeklatua(): Singleton patroia aplikatuz Teklatu bakarra bueltatzen du.
    - irakurriZenb(): Zenbaki bat teklatutik irakurtzen eta bueltatzen du.
    - irakurriString(): String bat teklatutik irakurtzen eta bueltatzen du.
    - irakurriChar(): Karaktere bat teklatutik irakurtzen eta bueltatzen du.
    - emanEnter(): Etendura bat sortzen du erabiltzailea "enter" botoiari eman arte.

FitxategiakIrakurri (EMA):

- Atributuak:
    - nireFitxeroakIrakurri: Singleton patroia aplikatzeko balio duen atributu estatikoa.
- Metodoak:
    - *FitxeroakIrakurri():* Klasearen eraikitzailea.
    - getNireFitxeroakIrakurri(): Singleton patroia aplikatuz FitxeroakIrakurri klase bakarra bueltatzen du.
    - fitxeroakErakuts(pFitxera:String): Sartutako fitxeroa irakurtzen du.
    - mapaIrakurri(pMapa: String): Egoeren mapa irakurtzen eta inprimatzen du kontsolan.

3.2.- Sekuentzia diagrama

A) HASIERAPENAK:

1. Saloia, hilerria eta protagonista EMA klaseak sortu.
2. “Hasiera.txt” fitxategia erakutsi eta hari dagokion matrizea kotsolan sortu.
3. Protagonistaren izena eskatzen eta gordetzen da.
4. “Azalpena.txt” fitxategia erakutsi.
5. emanEnter klasea exekutatu ostean, “aurkezpena.txt” erakutsi.

B) SALOIA ETA HILERRIA:

1. Eszenatokia hasieratu eta inprimatu:
   1.1. FitxeroakIrakurri klasean dagoen mapaIrakurri metodoari egoeraren mapa pasatu.
   1.2. Egoeran dauden NPCak (Not Playable Character) hasieratu.
2. Protagonistari egoera honetan duen hasierako posizioa esleitu.
3. Egoera honi dagokion ListaAkzioak sortu:

while(!listaAkzioak.bukatuSaloia()||!listaAkzioak.bukatuHilerria()){

4. Eszenatokia inprimatu.
5. Protagonistaren x eta y kordenatuak eskatu.
6. 2. puntuan sortutako ListaAkzioaren barnean dauden akzioen artean aukeratu eta akzio hori burutu:
   6.1. Zerrenda sortzen duten akzioak inprimatzen dira.
   6.2. Erabiltzaileak akzioen artean bat aukeratzeko zenbakiaAukeratu metodoaren bitartez zenbaki egoki bat irakurtzen du teklatutik.
   6.3. Akzioa aukeratu ostean, akzio hori betetzen da.
7. emanEnter() metodoa exekutatu ostean, pertsonaia dagoen kordenatutik ezabatzen da eta behar den kordenatu berrietan inprimatuko da.
8. eszenatokia bukatu eta inprimatu:
   8.1. FitxeroakIrakurri klasean dagoen mapaIrakurri metodoari egoera bukatuaren mapa pasatu.
   8.2. Teklatuan dagoen emanEnter() metodoa exekutatu.

}

C) BANKETXEA:

1. banketxea() metodoa exekutatu.
2. Banketxea, inbentarioa, listaGordelekuak eta protagonista EMA klaseak sortu.
3. eszenatokia hasieratu:
   3.1. FitxeroakIrakurri klasean dagoen mapaIrakurri metodoari egoeraren mapa pasatu.
   3.2. Egoeran dauden NPCak (Not Playable Character) hasieratu.
4. Protagonistari egoera honetan dauzkan hasierako kordenatuak esleitu.
5. Egoera honi dagokion ListaAkzioak sortu.
6. Predefinitutako gordelekuak daukagun ListaGordelekuan gorde eta gordelekuak inprimitu.
7. ListaEtsaiak klase bat sortu, bertan jadanik definitutako etsaiak sartu.
8. Banketxeak duenl Listae atributuan 7. pausuan sortutako lista esleitu.
9. Fitxeroak irakurri klasean fitxerokErakutsi metodoari "Banketxea/AzalpenaB.txt" pasatu, metodoak fitxeroa inprimatuko du.

while(!listaAkzioak.bukatuBanketxea){

10. eszenatokiaInprimatu.
11. Protagonistaren x eta y kordenatuak hartu.
12. Etsaiak banan banan eraso egin.
13. Etsaien bizitza inprimatu.
14. 5. puntuan sortutako ListaAkzioaren barnean dauden akzioen artean aukeratu eta akzio hori burutu:
    14.1. Zerrenda sortzen duten akzioak inprimatzen dira.
    14.2. Erabiltzaileak akzioen artean bat aukeratzeko zenbakiaAukeratu metodoaren bitartez zenbaki egoki bat irakurtzen du teklatutik.
    14.3. Akzioa aukeratu ostean, akzio hori betetzen da.

15. emanEnter() metodoa exekutatu ostean, pertsonaia dagoen kordenatutik esabatzen da eta behar den kordenatu berrietan inprimatuko da.
16. Etsai guztien bizitza<=0 bada AMAIERA.

}

C- AMAIERA:

1. Etsai guztien bizitza <=0 bada:
   1.1. Fitxeroak irakurri klasean fitxerokErakutsi metodoari "Amaiera.txt" pasatu, metodoak fitxeroa inprimatuko du.
   1.2. Teklatuaren emanEnter() metodoa exekutatu.

1.3. FitxeroakIrakurri klasean dagoen mapaIrakurri metodoari "Banketxea/Banketxea_Bukatu_Ondo.txt" fitxategia pasatu, honek fitxategia inprimatuko du.

2. Protagonistaren bizitza<=0 bada:
   2.1. Fitxeroak irakurri klasean fitxerokErakutsi metodoari "AmaieraTX.txt" pasatu, metodoak fitxeroa inprimatuko du.
   2.2. Teklatuaren emanEnter() metodoa exekutatu.
   2.3. Banketxea egoera berriro egiteko aukera eman:
      2.3.1. Aukera hau aukeratzen badu:
         2.3.1.1. BANKETXEA
      2.3.2. Erabiltzaileak ez badu aukeratzen:
         2.3.2.1. FitxeroakIrakurri klasean dagoen mapaIrakurri metodoari "Banketxea/Banketxea_Bukatu_Txarto.txt" fitxategia pasatu, honek fitxategia inprimatuko du.

4.- Proba kasuak – JUniten diseinua

Klase gehienen metodoen JUnitak oso sinpleak dira; ondorioz, ez du merezi aipatzea, baina horietatik pare bat aipatu eta azaldu behar ditugu:

- Metodo Aipagarrienak:

Gure ustez izan ditugun metodorik aipagarrienak metodoen barruan teklatutik irakurtzeko beharra dutenak izan dira. Assertak sartzeko metodoaren kode osoa (sysout komandoak ezik) kopiatu dugu, baina teklatuko balioa gordetzen zuen aldagaia aldatzen eta aldagai konstante bat sartzen. Mugitu() eta akzioaAukeratuEtaBurutu metodoetan() "teknika" hau inplementatu dugu.

Beste alde batetik Etsaia klasea duen eraso() metodoa assertTrue batekin egin behar izan dugu etsaileek random batekin jokalariaren PVren kantitate aldakor bat zutenez hasieratik zuen PV konparatu behar izan dugu, adibidez hasierako PV=100 badira honela egin behar izan dugu:

assertTrue(Protagonista.getNireProtagonista().getPv()<=100);

<=100 da eraso egiterakoan jokalariari min ez egitearen probabilitate txiki bat dagoelako.

- JUnitak ez duten metodoak:

Setterrak eta getterrak aparte (ez duelako merezi haiek JUnitak egitea), badago beste metodoren bat JUnitak ez dituena, adibidez Akzioa klasean

badago() metodoa. Metodo honek zure x edo y ardatzean dauden etsaien zerrenda bat ematen dizu. Banketxea egoeran etsaiak random batekin kokatzen direnez, ezin dezakegu jakin zein lekutan egongo diren, ondorioz ezin dezakegu Assert egin. Beste aldetik zerbait inprimatzen duten metodoak ere ez ezin ditugu JUnitekin frogatu, bakarrik exekutatzen eta kontsolan inprimatu behar dutena inprimatzen dutela ikusten.

Beste aldetik ListaAkzioa klasearen akzioaAukeratuEtaBurutu() metodoa ez dugu egin; izan ere, frogatu ahal diren bi kasuak, mugitu eta tiro egin, herentziak dira. Horregatik, haien JUnit propioetan frogatu ditugu.

5.- Salbuespenak

Salbuespenetarako hainbat salbuespen sortu genituen, orain aipatuko eta azalduko ditugu:

1. NotZenbakiEgokia: Hurrengo metodoak salbuespen bat erabiltzen edota botatzen dute:
    - public Etsaia returnPosizioan(int pPos) throws NotZenbakiEgokia{->Salbuespen hau goratuko du, sarutako balioren posizioa lista baino handiagoa bada.(ListaEtsaiak)
    - public void akzioaAukeratuEtaBurutu(int pEgoera) throws FileNotFoundException, IOException{->Akzioak aukeratzeko, zenbaki batzuk ditugu eta hauetako bat ez sakatzean salbuespen hau aktibatuko da(ListaAkzio klasean)
    - private int zenbakiaAukeratu(int pEgoera) throws NotZenbakiEgokia{->Salbuespen hau goratuko da, akzioak aukeratzeko zenbaki eremuak definituko dira(ListaAkzio klasean)
2. BalioEzEgokia: Hurrengo metodoak salbuespen hau erabiltzen edota botatzen dute:
    - private void tiroketa(ListaEtsaiak pLista)-> Do-While batekin konpondu arazoa, etsaia bat hil nahi dugunean, etsaia horren nick-a sartzen ez badugu salbuespena aktibatuko da. (TiroEgin klasean)
    - public void mugitu(){-> ‘W’,’A’,’S’,’D’ teklak ez sartzean, salbuespen hau aktibatuko da eta berriro ere tekla bat sartzeko aukera eman digu. (Mugitu klasean)
    - public boolean badago(Etsaia pEtsaia) throws BalioEzEgokia{-> Salbuespen hau goratzen du, listaetsaian etsaia aurkitzen ez bada (ListaEtsaiak)

- private int noranzkoaLortu() throws BalioEzEgokia{->Metodo hau salbuespen hau botatzen du, hau noranzkoa lortzeko erabiliko da eta mugitu metodoan sarttu behar diren balioak ez sartuz, salbuespena goratuko da. (Mugitu klasean)
- public Etsaia etsaiaBilatuIzenez(String pIzena) throws BalioEzEgokia{-> Salbuespen hau goratuko du, etsaia bat bilatzen eta bere izena ondo ez badugu sartzen salbuespena gortuko da. (ListaEtsaiak klasean)

Honetaz aparte, jadanik sortuta dauden salbuespenak hartu genituen, ondoren aipatuko eta azalduko ditugu:

1. FileNotFoundException: FItxategia aurkitzen ez denean exekutatzen da.
2. IOException: Sarrera edo irteeran errore bat gertatzen bada exekutatzen da.
3. InterruptedException: Eten bat sortzeko balio du.
4. NumberFormatException: Zenbaki bat eskatzerakoan jokalariak zenbaki bat ez den edozein karaktere sartzen duenean exekutatzen da. “Badakizu zenbakiak nola diren?” mezua inprimatzen du.

Hau tratatzeko salbuespen hauek ez exekutatu arte, hau da, guk onartutako balio bat sartu arte, egongo da “Scanner”etik balioak eskatzen.

6.- Inplementazioaren alde aipagarriak

Gure jokoaren kodea inplementatzerakoan hainbat zailtasun izan genituen, ondorioz, pare bat gauza alatu genituen. Proiektua pixkanaka pixkanaka eboluzionatzen joan da azkenengo bertsiora ailegatu arte. Hauek dira gure inplementazioaren alde aipagarrienak:

- ListaAkzioa klasean akzioaAukeratuEtaBurutu() metodoa daukagu, eta gure ustez metodo hau jokoaren alde garrantzitsuena da “The Ol’ Western”en mamia jokalariak hartu ahal dituen aukerak direlako. Metodo honek hasieran ea zein egoera zauden guk predefinitutako listaAkzioen artean bat aukeratzen du, listaAkzioakSortu(ListaAkzioa pLista) metodoari esker. Ondoren zerrenda horren akzioak pantailan inprimatzen ditu eta jokalariari horietako bat aukeratzeko aukera ematen dio, sartutako zenbakia ona dela erabaki ondoren, zenbakiaAukeratu() metodoari esker, programak jokalariak aukeratutako akzioa burutzen du.
- main(String args[]) metodoa ListaEgoerak klasean kokatuta dago. Bertan, jokoa exekutatzen da. Aurreko ataletan azalduta dagoenez, jokoa hiru

egoeratan banatuta dago; azkenengoa, tiroteoarena, berezia da eta bertan pertsonaia hil daiteke. Hau gertatzean, jokalariak berriro saia dezake egoera. Horren ondorioz, kodea banatzea erabaki genuen, eta bi tokitan kokatzea deia: Bata, ordenean, aurreko egoera eta gero. Bestea, aldiz, pertsonaia hiltzerakoan, jokalariak mapa errepikatzeko eskatzen badu.

- Aurreko puntuan esan dugun bezala, azken egoeraren kodea banatzea erabaki genuen, horregatik "main" metodoaren azken lerroak banketxea() metodoa exekutatzen du. Metodo honek Banketxea egoera hasieratzen du eta bertako elkarrekintzarako behar diren elementuak hartzen ditu, adibidez Inbentarioa eta banketxean dauden etsaien zerrenda (etsaien posizioa random batekin aukeratzen da). Sarreran esaten dugun bezala, gure lana etsai guztiak hiltzea da, hau betetzean jokoa amaitzen da baina jokalaria hiltzen badute ematen diogu aukera azken egoera hau hasieratik berriro errepikatzeko.

7.- Ondorioak

Proiektua egitea kriston lana izan da, baina honi esker hainbat ondorio atera ditugu:

1. Joko bat egitea ez da dirudien hain erraza.
2. Klase diagrama oso garrantzitsua da.
3. Taldean lan egiteko gaitasunak eskuratu ditugu.
4. Lauhilabetean zehar ikasitako kontzeptu guztiak(herentzia, salbuespenak, arrayList, etab.) hobeto ulertzen ditugu.
5. Lan asko, denbora tarte finitu batean egiteko gaitasunak eskuratu ditugu.

Honen ondorioz, esan dezakegu proiektua zerbait positiboa izan dela guretzat; izan ere, orain lehen baino gaitasun handiagoak ditugu eta gure arteko harremana hobetu da.

8.- Gehigarriak eta bibliografia

Proiektuaren hasieran Jon Anderrek jokoaren planteamendua, jokatzeko ahalmenaren ideia nagusia eta proiektuaren txantiloia ekarri zuen. Hortik aurrera, lauron artean ideiak ezartzen hasi ginen eta proiektua pixkanaka pixkana aldatzen joan zen gaur egun dugun bertsiora ailegatu arte.

Argi dago proiektu osoa ez dugula guk bakarrik egin; izan ere, hainbat lekutik informazioa atera dugu gure proiektuaren atal puntual batzuk era egokian inplementatzeko, baita klasean ez ditugun eman metodoen kudeaketa, adibidez

fitxategiak irakurtzeko erabili dugun FitxategiakIrakurri klasea. Leku horiek hauek dira nagusiki:

- eGelan jadanik inplementatuta dauden metodoak, adibidez dadoa.
- https://www.stackoverflow.com/
- https://docs.oracle.com/javase/7/docs/api/

9.- Kodea

Akzioa

```
package packproiektua;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.Iterator;

public class Akzioa {
  protected String izena;
  protected int ident;
  protected boolean kutxa;
  protected static boolean objetuak1=false;
  protected static boolean objetuak2=false;
  protected static int kont = 0;

  public Akzioa(String pIzena,int pIdent){
    this.izena = pIzena;
    this.ident=pIdent;
  }
  public int getIdent(){
    return this.ident;
  }

  public boolean getKutxa(){
    return this.kutxa;
  }
  public boolean kutxa(){
    boolean kutxa_da=true;
    return kutxa_da;
  }
  public void setObjetuak1(boolean pO){
    this.objetuak1=pO;
  }
  public void setObjetuak2(boolean pO){
    this.objetuak2=pO;
  }
  public void kont(){
    this.kont++;
  }

  public ListaEtsaiak berdinakDira(){
    int px = Protagonista.getNireProtagonista().getX();
    int py = Protagonista.getNireProtagonista().getY();
    Etsaia e = null;
    ListaEtsaiak l1 =
Banketxea.getNireBanketxea().lortuEtsaiakBanketxetik();
```

```
    ListaEtsaiak l2 = new ListaEtsaiak();
    Iterator<Etsaia> itr = l1.getIteradorea();
    while(itr.hasNext()){
      e = itr.next();
      int ex = e.getX();
      int ey = e.getY();
      if((px==ex)||(py==ey)){
        l2.etsaiaGehitu(e);
      }
    }
    return l2;
  }

  public void akzioaBurutu(int pEgoera) throws FileNotFoundException,
IOException{

    //int lag=Teklatua.getNireTeklatua().irakurriZenb();


    Boolean giltza=false; //Apaizarekin hitz egin eta gero true
bihurtuko da
    Protagonista p=Protagonista.getNireProtagonista();
    Saloia saloia = Saloia.getNireSaloia();
      Hilerria hilerria = Hilerria.getNireHilerria();
      Banketxea banketxea = Banketxea.getNireBanketxea();


    if(pEgoera==1){
      if(this.ident==1){

        int preX=p.getX();
          int preY=p.getY();
        saloia.deletePertsonaiaMatrizetik(preX, preY);
        p.posizioazAldatu(12, 4);
        saloia.setPertsonaiaMatrizean();
          saloia.eszenatokiaInprimatu();

        System.out.println("Tabernariarengana hurbildu zara eta
berarekin hitz egiten saiatu zara...");
        if(dialogoaBurutu()){

FitxeroakIrakurri.fitxeroaErakutsi("Saloia/Tabernaria_T.txt");
          ListaAkzioa listaAk = new ListaAkzioa();
          listaAk.setKutxaT();
          saloia.kutxaAgertu();
        }
        else{

FitxeroakIrakurri.fitxeroaErakutsi("Saloia/Tabernaria_F.txt");
        }
      }
      else if(this.ident==2){

        int preX=p.getX();
          int preY=p.getY();
        saloia.deletePertsonaiaMatrizetik(preX, preY);
        p.posizioazAldatu(4, 16);
        saloia.setPertsonaiaMatrizean();
          saloia.eszenatokiaInprimatu();
```

```
        System.out.println("Prostitutarengana hurbildu zara eta
berarekin hitz egiten saiatu zara...");
        if(dialogoaBurutu()){

FitxeroakIrakurri.fitxeroaErakutsi("Saloia/Prostituta_T.txt");
        }
        else{

FitxeroakIrakurri.fitxeroaErakutsi("Saloia/Prostituta_F.txt");
        }
      }
      else if(this.ident==3){

        int preX=p.getX();
          int preY=p.getY();
        saloia.deletePertsonaiaMatrizetik(preX, preY);
        p.posizioazAldatu(7, 7);
        saloia.setPertsonaiaMatrizean();
          saloia.eszenatokiaInprimatu();

        System.out.println("Gizon zaharrarenga hurbidu zara...");
        FitxeroakIrakurri.fitxeroaErakutsi("Saloia/GizonZaharra.txt");
      }
      else if(this.ident==4){

        int preX=p.getX();
          int preY=p.getY();
        saloia.deletePertsonaiaMatrizetik(preX, preY);
        p.posizioazAldatu(3, 3);
        saloia.setPertsonaiaMatrizean();
          saloia.eszenatokiaInprimatu();

        System.out.println("Kutxagogorrera hurbildu zara eta
irekitzeko gako bat behar duela ikusten duzu...");
        int gakoa=zenbakiaLortu();
        //String kontra=Integer.toString(gakoa);
        //if(kontra=="1830"){
        if(gakoa==1830){

FitxeroakIrakurri.fitxeroaErakutsi("Saloia/Kutxagogorra.txt");
          ListaAkzioa l = new ListaAkzioa();
          l.pasatuSaloitikHilerrira(); //hurrengo egoerara pasatuko da
        }
        else{
          System.out.println("Kutxagogorra irekitzen saiatu zara baina
ez da ezer gertatu...");
        }
      }
    }
    else if(pEgoera==2){
      if(this.ident==1){

        int preX=p.getX();
          int preY=p.getY();
        hilerria.deletePertsonaiaMatrizetik(preX, preY);
        p.posizioazAldatu(8, 2);
        hilerria.setPertsonaiaMatrizean();
        hilerria.eszenatokiaInprimatu();

        System.out.println("Ehorzlearengana hurbildu zara eta
berarekin hitz egiten saiatu zara...");
```

```
        FitxeroakIrakurri.fitxeroaErakutsi("Hilerria/Ehorzlea.txt");
      }
      else if(this.ident==2){

        int preX=p.getX();
          int preY=p.getY();
        hilerria.deletePertsonaiaMatrizetik(preX, preY);
        p.posizioazAldatu(7, 15);
        hilerria.setPertsonaiaMatrizean();
        hilerria.eszenatokiaInprimatu();

        System.out.println("Apaizarengana hurbildu zara eta berarekin
hitz egiten saiatu zara...");
        FitxeroakIrakurri.fitxeroaErakutsi("Hilerria/Apaiza.txt");
        giltza=true;
        ListaAkzioa l = new ListaAkzioa();
        l.setElizaT();
      }
      else if(this.ident==3){

        int preX=p.getX();
          int preY=p.getY();
        hilerria.deletePertsonaiaMatrizetik(preX, preY);
        p.posizioazAldatu(7, 12);
        hilerria.setPertsonaiaMatrizean();
        hilerria.eszenatokiaInprimatu();

        ListaAkzioa l = new ListaAkzioa();
            if(l.getEliza()){

FitxeroakIrakurri.fitxeroaErakutsi("Hilerria/Eliza_T.txt");
              //ListaEgoerak.getNireListaEgoerak().egoeraEguneratu(3);
//hurrengo egoerara pasatuko da

              l.pasatuHilerritikBanketxera(); //hurrengo egoerara
pasatuko da
            }
            else{

FitxeroakIrakurri.fitxeroaErakutsi("Hilerria/Eliza_F.txt");
            }
      }
    }
    else if(pEgoera==3){

      if(this.ident==1){
        TiroEgin t=new TiroEgin();
        t.tiroEgin();
      }
      else if(this.ident==2){
        if(!objetuak1){
          System.out.println("Lehen zenuen bizitza hurrengoa da:");
          System.out.println("Bizitza: "+p.getPv());
          System.out.println("");
          p.objetuaErabili("Pitia");
          System.out.println("Pitia erabili duzu eta zure bizitza
hurrengoa da:");
          System.out.println("Bizitza berria: "+p.getPv());
        this.setObjetuak1(true);
        }
        else{
```

```
            System.out.println("Pitia ez dago zure inbentarioan");
          }
        }
        else if(this.ident==3){
          if(!objetuak2){
            System.out.println("Lehen zenuen bizitza hurrengoa da:");
            System.out.println("Bizitza: "+p.getPv());
            System.out.println("");
            p.objetuaErabili("Kapela");
            System.out.println("Kapela erabili duzu eta zure bizitza
hurrengoa da:");
            System.out.println("Bizitza berria: "+p.getPv());
            this.setObjetuak2(true);
            }
            else{
              System.out.println("Kapela ez dago zure inbentarioan");
            }
        }
        else if(this.ident==4){
          if(this.kont<=1){
            System.out.println("Lehen zenuen bizitza hurrengoa da:");
            System.out.println("Bizitza: "+p.getPv());
            System.out.println("");
            p.objetuaErabili("Likorea");
            System.out.println("Likorea erabili duzu eta zure bizitza
hurrengoa da:");
            System.out.println("Bizitza berria: "+p.getPv());
            this.kont();
            }
            else{
              System.out.println("Likorea ez dago zure inbentarioan");
            }
        }

        else if(this.ident==5){
          System.out.println("Norantz nahi duzu mugitu?");
          System.out.println("> W < gorantz joateko");
          System.out.println("> A < ezkerrerantz joateko");
          System.out.println("> S < beherentzat joateko");
          System.out.println("> D < eskuineranz joateko");
          Mugitu m=new Mugitu();
          m.mugitu();
        }
      }

    }

    private boolean dialogoaBurutu(){
      Boolean burutu=false;
      Dadoa d=new Dadoa(6);
      d.bota();
      /*

if(Protagonista.getNireProtagonista().getCar()+d.getGoikoAldea()>=6) {
        burutu=true;
      }*/
      if(d.getGoikoAldea()>=3) {
        burutu=true;
      }
      return burutu;
    }
```

```
  public void akzioaInprimatu(){
    System.out.println(this.izena);
  }
  public void setIdent(int pIdentitatea){
    this.ident=pIdentitatea;
  }

  private int zenbakiaLortu(){
    boolean lortuta=false;
    try{
      int lag=Teklatua.getNireTeklatua().irakurriZenb();
      lortuta=true;
      return lag;
    }
    catch(NumberFormatException lag){
      System.out.println("Benetan badakizu zenbakiak nola diren?");
      return this.zenbakiaLortu();
    }
  }
}
```

BalioEzEgokia

```
package packproiektua;

public class BalioEzEgokia extends Exception{

  public BalioEzEgokia(){
    super();
  }
}
```

Banketxea (EMA)

```
package packproiektua;

import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.Iterator;
import java.util.ArrayList;

public class Banketxea extends Egoera{

  private char[][] matrizea;
  private static Banketxea nireBanketxea = null;
  private ListaEtsaiak listae;
  private ListaGordelekuak listagor;

  private Banketxea(String pDeskribapena){
    super(pDeskribapena);
    this.matrizea = new char[20][20];
    this.idEgoera=3;
    this.listagor=ListaGordelekuak.getNireListaGordelekuak();
    this.listae=new ListaEtsaiak();
  }
  public static Banketxea getNireBanketxea(){
    if(nireBanketxea==null){
      nireBanketxea = new Banketxea("Banketxea");
    }
    return nireBanketxea;
```

```
  }

  public ListaGordelekuak gordelekuakSortu(){
    ListaGordelekuak lista =
ListaGordelekuak.getNireListaGordelekuak();
    return lista;

  }
  public ListaAkzioa getLista(){
    return this.lista.listaAkzioakSortu(3);

  }

  public void eszenatokiaHasieratu() throws FileNotFoundException,
IOException{

this.matrizea=FitxeroakIrakurri.mapaIrakurri("Banketxea/Banketxea.txt"
);
    this.pertsonaiakHasieratu();
  }

  public void eszenatokiaBukatuOndo() throws FileNotFoundException,
IOException{

this.matrizea=FitxeroakIrakurri.mapaIrakurri("Banketxea/Banketxea_Buka
tuta_ONDO.txt");
    System.out.println();
    System.out.println();
    for (int i=0;i<20;i++) {
        for (int j=0;j<20;j++) {
          System.out.print(this.matrizea[i][j]+" ");
        }
        System.out.println();
    }
    System.out.println();
    System.out.println();
  }
  public void eszenatokiaBukatuTxarto() throws FileNotFoundException,
IOException{

this.matrizea=FitxeroakIrakurri.mapaIrakurri("Banketxea/Banketxea_Buka
tuta_TXARTO.txt");
    System.out.println();
    System.out.println();
    for (int i=0;i<20;i++) {
        for (int j=0;j<20;j++) {
          System.out.print(this.matrizea[i][j]+" ");
        }
        System.out.println();
    }
    System.out.println();
    System.out.println();
  }

  public void eszenatokiaInprimatu(){
    System.out.println();
    System.out.println();
    for (int i=0;i<20;i++) {
        for (int j=0;j<20;j++) {
          System.out.print(this.matrizea[i][j]+" ");
        }
```

```
            System.out.println();
        }
        System.out.println();
        System.out.println();
    }


    public void gordelekuakInprimatu(ListaGordelekuak lista1){
        int x = 0;
        int y = 0;
        Gordelekua gordelekua = null;
        Iterator<Gordelekua> itr = lista1.getIteradorea();
        while(itr.hasNext()){
            gordelekua = itr.next();
            x = gordelekua.getX();
            y = gordelekua.getY();
            matrizea[x][y]='%';
        }
    }
    public void setEtsaiakMatrizean(int rx, int ry, String pIzena){
        matrizea[rx][ry]=pIzena.charAt(0);
    }

    public void setPertsonaiaMatrizean(){
        Protagonista p = Protagonista.getNireProtagonista();
        int x= p.getX();
        int y=p.getY();
        matrizea[x][y]='#';
    }

    public ListaEtsaiak lortuEtsaiakBanketxetik(){
        return this.listae;
    }
    public void sartuEtsaiakBanketxean(ListaEtsaiak pLista){
        this.listae=pLista;
    }
    public void deletePertsonaiaMatrizetik(int pX, int pY){
        matrizea[pX][pY]=' ';
    }
    public char matrizekoBalioa(int x, int y){
        return matrizea[x][y];
    }
    public void etsaiaHildaMatrizean(int pX, int pY){
        matrizea[pX][pY]='X';
    }
    private void pertsonaiakHasieratu(){



    }
}
```

Dadoa

```
package packproiektua;

import java.util.Random;

public class Dadoa {
    //atributuak
    private int goikoAldea;
```

```
    private int aldeKopurua;

    //metodo eraikitzaileak
    public Dadoa(int pAldeKopurua){
      aldeKopurua=pAldeKopurua;
    }

    //getters && setters

    public void setGoikoAldea(int pGoikoAldea){
      this.goikoAldea=pGoikoAldea;
    }

    public int getGoikoAldea(){
      return this.goikoAldea;
    }

    public int getAldeKopurua(){
      return this.aldeKopurua;
    }


    public void bota(){
      Random zenbakiRandomak=new Random();
      int egungoBalioa;
      egungoBalioa=zenbakiRandomak.nextInt(this.getAldeKopurua());
      this.setGoikoAldea(egungoBalioa);
    }

}
```

Egoera

```
package packproiektua;

import java.util.ArrayList;
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.Iterator;

public abstract class Egoera {
  protected char[][] matrizea;
  private String deskribapena;
  protected ListaAkzioa lista;
  protected int idEgoera;
  private static Egoera nireEgoera = null;
  protected ListaGordelekuak lista1;

  public Egoera(String pDeskribapena){
    this.deskribapena = pDeskribapena;
    this.lista= new ListaAkzioa();
    this.lista1=ListaGordelekuak.getNireListaGordelekuak();
  }



  public void egoeraInprimatu(){
    System.out.println(this.deskribapena);
  }
```

```
  /*
  public void ezsenatokiaInprimatu(Egoera pEgoera) throws
FileNotFoundException, IOException{
    if(pEgoera instanceof Hilerria){
      FitxeroakIrakurri.fitxeroaErakutsi("Hilerria");
    }
    else{
      if(pEgoera instanceof Saloia){
        FitxeroakIrakurri.fitxeroaErakutsi("Saloia");
      }
      else{
        FitxeroakIrakurri.fitxeroaErakutsi("Banketxea");
      }
    }
  }*/

  public int egungoEgoeraLortu(){
    return this.idEgoera;
  }

  /*
  public void printeatuEgoerarenAukerak() throws
FileNotFoundException, IOException{
    ListaAkzioa l= new ListaAkzioa();
    if(this.idEgoera==1){
      l.listaAkzioakSortu(nireEgoera);
    }
    else{
      if(this.idEgoera==2){
        l.listaAkzioakSortu(nireEgoera);
      }
      else{
        if(this.idEgoera==3){
          l.listaAkzioakSortu(nireEgoera);
        }
      }
    }
    l.akzioaAukeratuEtaBurutu(this.idEgoera);
  }

  */

  public void hasieratu(){
    this.lista.clear();
    ListaAkzioa l= new ListaAkzioa();
    l.listaAkzioakSortu(idEgoera);
    this.lista=l;
  }

}
```

Etsaia

```
package packproiektua;

public class Etsaia extends Pertsonaia{

  private static int atq=5;

  public Etsaia(String pIzena,int pX,int pY){
    super(pIzena);
    this.pv = 100;
```

```
    this.izena=pIzena;
    this.x=pX;
    this.y=pY;
  }
  public int getPv(){
    return this.pv;
  }

  public void eraso(){
    Protagonista p = Protagonista.getNireProtagonista();
    //En este metodo, lo qeu haria seria, si nuestro personaje esta
cubierto, el etsaia nos va a quitar 0 de vida.
    //Si nuestro personaje no esta cubierto, que nos quite (5 o 10) de
vida
    if(p.estalita()){
      //Ez egin ezer
    }
    else{
      Dadoa d=new Dadoa(12);
      d.bota();
      int ga=d.getGoikoAldea();
      if(ga==1 || ga==2 || ga==3){
        p.setPv(p.getPv()-this.atq*1);
      }
      else if(ga==4 || ga==5 || ga==6){
        p.setPv(p.getPv()-this.atq*2);
      }
      else if(ga==7 || ga==8){
        p.setPv(p.getPv()-this.atq*3);
      }
      else if(ga==9){
        p.setPv(p.getPv()-this.atq*4);
      }
      else if(ga==11){
        p.setPv(p.getPv()-this.atq*5);
      }
      else if(ga==12){
        p.setPv(p.getPv()-this.atq*6);
      }
    }
  }
  public boolean hilda(){
    boolean hilda = false;
    if(this.pv <=0){
      hilda = true;
      Banketxea.getNireBanketxea().etsaiaHildaMatrizean(x, y);
    }
    return hilda;
  }
  public String getIzena(){
    return this.izena;
  }
  public int getX(){
    return this.x;
  }
  public int getY(){
    return this.y;
  }
  public void setBizitza(int pV){
    this.pv=pV;
  }
```

```
}
```

FitxeroakIrakurri (EMA)

```
package packproiektua;

import java.io.FileNotFoundException;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
//
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.io.FileNotFoundException;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.Scanner;

public class FitxeroakIrakurri {

  private static FitxeroakIrakurri nireFitxeroakIrakurri = null;

  private FitxeroakIrakurri(){

  }

  public static FitxeroakIrakurri getNireFitxeroakIrakurri(){
    if(nireFitxeroakIrakurri==null){
      nireFitxeroakIrakurri=new FitxeroakIrakurri();
    }
    return nireFitxeroakIrakurri;
  }

  public static void fitxeroaErakutsi(String pFitxeroa) throws
FileNotFoundException, IOException{
    String katea;
    //FileReader f=new FileReader(pFitxeroa,"UTF-8");
    InputStreamReader f=new InputStreamReader(new
FileInputStream("./fitxategiak/" + pFitxeroa), "UTF-8");
    BufferedReader b=new BufferedReader(f);
    b.skip(1);
    while((katea=b.readLine())!=null){
      System.out.println(katea);
    }
    b.close();
  }

  public static char[][] mapaIrakurri(String pFitxeroa) throws
FileNotFoundException, IOException{
    char[][] matrizea = new char[20][20];
    //FileReader f=new FileReader(pFitxeroa,"UTF-8");
    InputStreamReader f=new InputStreamReader(new
FileInputStream("./fitxategiak/" + pFitxeroa), "UTF-8");
    BufferedReader b=new BufferedReader(f);
    //b.skip(1);
    for (int i=0;i<20;i++) {
```

```
        for (int j=0;j<20;j++) {
          char c = (char)b.read();
          matrizea[i][j]=c;

        }
    }
    b.close();
    return matrizea;
  }
}
```

Gordelekua

```
package packproiektua;

public class Gordelekua {

  private int x;
  private int y;

  public Gordelekua(int pX, int pY){
    this.x = pX;
    this.y = pY;
  }

  public int getX(){
    return this.x;
  }

  public int getY(){
    return this.y;
  }
}
```

Hilerria (EMA)

```
package packproiektua;

import java.io.FileNotFoundException;
import java.io.IOException;

public class Hilerria extends Egoera{

  private static Hilerria nireHilerria = null;

  private Hilerria(String pDeskribapena){
    super(pDeskribapena);
    this.idEgoera=2;
    this.matrizea = new char[20][20];
  }
  public static Hilerria getNireHilerria(){
    if(nireHilerria==null){
      nireHilerria = new Hilerria("Hilerria");
    }
    return nireHilerria;
  }
  public ListaAkzioa getLista(){
    return this.lista.listaAkzioakSortu(2);

  }
```

```
  public void eszenatokiaHasieratu() throws FileNotFoundException,
IOException{

this.matrizea=FitxeroakIrakurri.mapaIrakurri("Hilerria/Hilerria.txt");
    this.pertsonaiakHasieratu();
  }

  public void eszenatokiaBukatu() throws FileNotFoundException,
IOException{

this.matrizea=FitxeroakIrakurri.mapaIrakurri("Hilerria/Hilerria_Bukatu
ta.txt");
    System.out.println();
    System.out.println();
    for (int i=0;i<20;i++) {
        for (int j=0;j<20;j++) {
          System.out.print(this.matrizea[i][j]+" ");
        }
        System.out.println();
    }
    System.out.println();
    System.out.println();
  }

  public void eszenatokiaInprimatu(){
    System.out.println();
    System.out.println();
    for (int i=0;i<20;i++) {
        for (int j=0;j<20;j++) {
          System.out.print(this.matrizea[i][j]+" ");
        }
        System.out.println();
    }
    System.out.println();
    System.out.println();
  }

  public void setPertsonaiaMatrizean(){
    Protagonista p = Protagonista.getNireProtagonista();
    int x= p.getX();
    int y=p.getY();
    matrizea[x][y]='#';
  }

  public void deletePertsonaiaMatrizetik(int pX, int pY){
    matrizea[pX][pY]=' ';
  }
  public char matrizekoBalioa(int x, int y){
    return matrizea[x][y];
  }

  private void pertsonaiakHasieratu(){

        matrizea[6][13]='^';
        matrizea[7][13]='^';
        matrizea[8][13]='^';
        matrizea[8][16]='A';
        matrizea[7][3]='E';

    }
```

```
}
```

Inbentarioa (EMA)

```
package packproiektua;

import java.util.ArrayList;
import java.util.Iterator;

public class Inbentarioa {
  private ArrayList<Objetua>lista;
  private static Inbentarioa nireInbentarioa = null;

  private Inbentarioa(){
    this.lista = new ArrayList<Objetua>();
  }
  public static Inbentarioa getNireInbentarioa(){
    if(nireInbentarioa==null){
      nireInbentarioa=new Inbentarioa();
    }
    return nireInbentarioa;
  }


  private Iterator<Objetua> getIteradorea(){
    return this.lista.iterator();
  }
  private void ObjetuaGehitu(Objetua pObjetua){
    this.lista.add(pObjetua);
  }

  public Inbentarioa inbentarioaSortu(){
    Inbentarioa inb = Inbentarioa.getNireInbentarioa();

    Kapela obj = new Kapela();
    inb.ObjetuaGehitu(obj);
    Pitia obj1 = new Pitia();
    inb.ObjetuaGehitu(obj1);
    Likorea obj2 = new Likorea();
    inb.ObjetuaGehitu(obj2);
    return inb;
  }

  private Objetua bilatuObjetuaIzenez(String pString){
    Iterator<Objetua>itr=this.getIteradorea();
    Objetua o=null;
    boolean topatuta=false;
    while(itr.hasNext() && !topatuta){
      o=itr.next();
      if(pString.equals(o.getIzena())){
        topatuta=true;
      }
    }
    return o;
  }

  public void objetuaErabili(String pObjetua){
    boolean aurkitua=false;
    Objetua objetua=this.bilatuObjetuaIzenez(pObjetua);
```

```
    if((objetua.getIzena().equals(pObjetua)) && (objetua instanceof
Kapela)){
      aurkitua = true;
      Kapela kap=new Kapela();
      kap.objektuaErabili();
    }
    else if((objetua.getIzena().equals(pObjetua)) && (objetua
instanceof Likorea)){
      aurkitua = true;
      Likorea lik=new Likorea();
      lik.objektuaErabili();
    }
    else if((objetua.getIzena().equals(pObjetua)) && (objetua
instanceof Pitia)){
            aurkitua = true;
            Pitia piti=new Pitia();
            piti.objektuaErabili();
    }
    else if(!aurkitua){
      System.out.println("Ez da zure objetua aurkitu inbentarioan");
    }
  }
}
```

Kapela

```
package packproiektua;

public class Kapela extends Objetua{


  public Kapela(){
    super("Kapela");

  }

  public void objektuaErabili(){
    Protagonista p=Protagonista.getNireProtagonista();
    p.setPv(p.getPv()+50);
  }

}
```

Likorea

```
package packproiektua;

public class Likorea extends Objetua{

  public Likorea(){
    super("Likorea");

  }
  public void objektuaErabili(){
    Protagonista p=Protagonista.getNireProtagonista();
    p.setPv(p.getPv()+100);

  }
}
```

ListaAkzioa

```
package packproiektua;

import java.util.ArrayList;
import java.util.Iterator;
import java.io.FileNotFoundException;
import java.io.IOException;

public class ListaAkzioa{

  private ArrayList<Akzioa> lista;
  private static boolean kutxa=false;
  private static boolean eliza=false;
  private static boolean mugitu=false;
  private static boolean mugitu1=false;
  private static boolean pasatuSaloitikHilerrira=false;
  private static boolean pasatuHilerritikBanketxera=false;
  private static boolean pasatuBanketxetikAmaierara=false;

  public ListaAkzioa(){
    this.lista = new ArrayList<Akzioa>();
  }

  public void setKutxaT(){
    this.kutxa=true;
  }

  public void setElizaT(){
    this.eliza=true;
  }

  public void setMugitu(boolean pM){//Exception
    this.mugitu=pM;
  }

  public void setMugitu1(boolean pM){//Teklatua
    this.mugitu1=pM;
  }
  public boolean getKutxa(){
    return this.kutxa;
  }

  public boolean getEliza(){
    return this.eliza;
  }

  public boolean getMugitu(){
    return this.mugitu;
  }

  public boolean getMugitu1(){
    return this.mugitu1;
    }

  public void pasatuSaloitikHilerrira(){
    this.pasatuSaloitikHilerrira=true;
  }

  public void pasatuHilerritikBanketxera(){
    this.pasatuHilerritikBanketxera=true;
```

```
}

public void pasatuBanketxetikAmaierara(){
  this.pasatuBanketxetikAmaierara=true;
}

public boolean bukatutaSaloia(){
  return this.pasatuSaloitikHilerrira;
}

public boolean bukatutaHilerria(){
  return this.pasatuHilerritikBanketxera;
}

public boolean bukatutaBanketxea(){
  return this.pasatuBanketxetikAmaierara;
}

public void berrizHasieratuBanketxea(){
  this.pasatuBanketxetikAmaierara=false;
}

private Iterator<Akzioa> getIteradorea(){
  return this.lista.iterator();
}

public void printeatuAkzioa(){
  Akzioa akzioa = null;
  Iterator<Akzioa> itr = this.getIteradorea();
  while(itr.hasNext()){
    akzioa = itr.next();
    akzioa.akzioaInprimatu();
  }
}

public ListaAkzioa listaAkzioakSortu(int pEgoera){
  ListaAkzioa l=new ListaAkzioa();
  Akzioa a=null;
  if(pEgoera == 2){
    a=new Akzioa("Ehorzlearekin hitz egin",1);
    l.akzioaGehitu(a);
    a=new Akzioa("Apaizarekin hitz egin",2);
    l.akzioaGehitu(a);
    a=new Akzioa("Elizan sartu",3);
    l.akzioaGehitu(a);
  }
  else if(pEgoera == 1){
      a=new Akzioa("Tabernariarekin hitz egin",1);
      l.akzioaGehitu(a);
      a=new Akzioa("Prostitutarekin hitz egin",2);
      l.akzioaGehitu(a);
      a=new Akzioa("Gizon zaharrarekin",3);
      l.akzioaGehitu(a);
      a=new Akzioa("Kutxagogorrera hurbildu",4);
      l.akzioaGehitu(a);
  }
  else if (pEgoera==3){ //Banketxea
      a=new Akzioa("Tiro egin",1);
      l.akzioaGehitu(a);
      a=new Akzioa("Pitia erabili",2);
      l.akzioaGehitu(a);
```

```
        a=new Akzioa("Kapela erabili",3);
        l.akzioaGehitu(a);
        a=new Akzioa("Likorea erabili",4);
        l.akzioaGehitu(a);
        a=new Akzioa("Mugitu",5);
        l.akzioaGehitu(a);
    }

    return l;
  }

  public void akzioaAukeratuEtaBurutu(int pEgoera) throws
FileNotFoundException, IOException{
    try{
      String izena=Protagonista.getNireProtagonista().getIzena();

      if(pEgoera==1){
        //this.lista.listaAkzioakSortu(1);
        System.out.println(izena + ", zer egin nahi duzu?");
        System.out.println("1) Tabernariarekin hitz egin");
        System.out.println("2) Prostitutarekin hitz egin");
        System.out.println("3) Gizon zaharrarekin hitz egin");
        if(this.kutxa){
          System.out.println("4) Kutxagogorrera joan");
        }
      }
      else if(pEgoera==2){
        //this.lista.listaAkzioakSortu(2); esto va en el main
        System.out.println(izena + ", zer egin nahi duzu?");
        System.out.println("1) Ehorzlearekin hitz egin");
        System.out.println("2) Apaizarekin hitz egin");

          System.out.println("3) Elizan sartu");


      }
      else if(pEgoera==3){
        //this.listaAkzioakSortu(3);
        System.out.println(izena + ", zer egin nahi duzu?");
        System.out.println("1) Tiro egin");
        System.out.println("2) Pitia erabili");
        System.out.println("3) Kapela erabili");
        System.out.println("4) Likorea erabili");
        System.out.println("5) Mugitu");

      }



      int lag=this.zenbakiaAukeratu(pEgoera);
    //  if(lag==0){
    //    ListaEtsaiak pL =
Banketxea.getNireBanketxea().lortuEtsaiakBanketxetik();
    //    pL.etsaiakHilBETATESTER();//frogak egiteko
    //  }
      //int lag=Teklatua.getNireTeklatua().irakurriZenb();
      Akzioa a=null;
      boolean aurkitua=false;
      Iterator<Akzioa>itr=this.getIteradorea();
      while(itr.hasNext()&&!aurkitua){
        a=itr.next();
```

```
        if(a.getIdent()==lag){
          aurkitua=true;
        }
      }
      if(!aurkitua){
        System.out.println("Ez da aurkitu"); //testu hau agertzen da
        throw new NotZenbakiEgokia();
      }
      a.akzioaBurutu(pEgoera);
      //Darle al enter
    }
    catch(NotZenbakiEgokia lag){
      System.out.println("Mesedez, ez izan gringo eta sartu balio duen
zenbaki bat...");
    }
    catch(NumberFormatException lag){
      if(this.mugitu){
        System.out.println("Badakizu nola diren zenbakiak?");
      }
      this.setMugitu(false);

    }
  }

  private void akzioaGehitu(Akzioa pAkzioa){
    this.lista.add(pAkzioa);
  }


  public void clear(){
    this.lista.clear();
  }

  private int zenbakiaAukeratu(int pEgoera) throws NotZenbakiEgokia{
    int lag=Teklatua.getNireTeklatua().irakurriZenb();

    if(pEgoera==1){
      if(!this.kutxa){
        if((lag<0)||(lag>3)){
          throw new NotZenbakiEgokia();
        }
      }

      if(this.kutxa){
        if((lag<0)||(lag>4)){
          throw new NotZenbakiEgokia();
        }
      }
    }
    else if(pEgoera==2){
      if((lag<0)||(lag>3)){
        throw new NotZenbakiEgokia();
      }
    }
    else if(pEgoera==3){
      if((lag<0)||(lag>5)){
        throw new NotZenbakiEgokia();
      }
    }
```

```
    if((lag<0)||(lag>5)){
      throw new NotZenbakiEgokia();
    }
    return lag;
  }
}
```

ListaEgoerak (EMA)

```
package packproiektua;

import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Iterator;

public class ListaEgoerak {
  private ArrayList<Egoera> lista;
  private static ListaEgoerak nireListaEgoerak = null;

  private ListaEgoerak(){
    this.lista = new ArrayList<Egoera>();
  }
  public static ListaEgoerak getNireListaEgoerak(){
    if(nireListaEgoerak == null){
      nireListaEgoerak = new ListaEgoerak();
    }
    return nireListaEgoerak;
  }
  public Egoera egoeraEguneratu(int pZenb){
    Egoera e=null;
    int lag=pZenb;
    Iterator<Egoera>itr=this.getIteradorea();
    while(itr.hasNext()&&lag>0){
      e=itr.next();
      lag--;
    }
    return e;

  }
  private Iterator<Egoera> getIteradorea(){
    return this.lista.iterator();
  }


  public void hasieratuEgoerak(){
    Iterator<Egoera>itr=this.getIteradorea();
    Egoera e=null;
    while(itr.hasNext()){
      e=itr.next();
      e.hasieratu();
    }
  }

  public static void main(String[] args) throws FileNotFoundException,
IOException, InterruptedException, BalioEzEgokia{
    /*

     *    JOKOAREN BALIOAK HASIERATU
     *
     */
```

```
            int preX;
            int preY;
            ListaAkzioa listaAkz = new ListaAkzioa();
            Saloia saloia = Saloia.getNireSaloia();
            Hilerria hilerria = Hilerria.getNireHilerria();
            Banketxea banketxea = Banketxea.getNireBanketxea();
            Inbentarioa inb = Inbentarioa.getNireInbentarioa();
            saloia.jokoaHasieratu();
            Protagonista p = Protagonista.getNireProtagonista();
        /*
         *
         *     SALOIA
         *
         */    //SALOIKO BALIOAK HASIERATU

            saloia.eszenatokiaHasieratu();
            p.hasierakoPosizioa(1);
            ListaAkzioa listaAkzS=listaAkz.listaAkzioakSortu(1);

            //SALOIA EGOERA
            while(!listaAkz.bukatutaSaloia()){
            saloia.eszenatokiaInprimatu();
            preX=p.getX();
            preY=p.getY();
            listaAkzS.akzioaAukeratuEtaBurutu(1);
            Teklatua.getNireTeklatua().emanEnter();
            saloia.deletePertsonaiaMatrizetik(preX, preY);
            saloia.setPertsonaiaMatrizean();
            }

            //EGOERAZ ALDATU
            saloia.eszenatokiaBukatu();
            Teklatua.getNireTeklatua().emanEnter();
        /*
         *
         *     HILERRIA
         *
         */
            hilerria.eszenatokiaHasieratu();

            p.hasierakoPosizioa(2);
            ListaAkzioa listaAkzH=listaAkz.listaAkzioakSortu(2);


            while(!listaAkz.bukatutaHilerria()){
            hilerria.eszenatokiaInprimatu();
            preX=p.getX();
            preY=p.getY();
            listaAkzH.akzioaAukeratuEtaBurutu(2);
            Teklatua.getNireTeklatua().emanEnter();
            hilerria.deletePertsonaiaMatrizetik(preX, preY);
            hilerria.setPertsonaiaMatrizean();
            }

            hilerria.eszenatokiaBukatu();
            banketxea();
    }

       public static void banketxea() throws FileNotFoundException,
    IOException, InterruptedException, BalioEzEgokia{
          //BANKETXEA//
```

```
      int preX;
        int preY;
        ListaAkzioa listaAkz = new ListaAkzioa();
        Banketxea banketxea = Banketxea.getNireBanketxea();
        Inbentarioa inb = Inbentarioa.getNireInbentarioa();
        Protagonista p = Protagonista.getNireProtagonista();


        ListaGordelekuak l1 =
ListaGordelekuak.getNireListaGordelekuak();
        //Banketxearen balioak hasieratu//
        banketxea.eszenatokiaHasieratu();
        p.hasierakoPosizioa(3);
        ListaAkzioa listaAkzB=listaAkz.listaAkzioakSortu(3);
        l1.gordelekuakSortu();
        banketxea.gordelekuakInprimatu(l1);
        inb.inbentarioaSortu();
        ListaEtsaiak listae = new ListaEtsaiak();


        Teklatua.getNireTeklatua().emanEnter();

        listae.etsaiakSortu();
        banketxea.sartuEtsaiakBanketxean(listae);


        while(!listaAkz.bukatutaBanketxea()){
          banketxea.eszenatokiaInprimatu();
          preX=p.getX();
          preY=p.getY();
          System.out.println("+++++ ETSAIEN TXANDA +++++");

          listae.eraso();
          if(p.getPv()<=0){
            System.out.println("Hilda zaude.");
          }
          else{
          System.out.println("Etsaien erasoaren ondoren, zure bizitza
"+ p.getPv() +" da");
          System.out.println();
          System.out.println("+++++ ZURE TXANDA +++++");
          listae.etsaienBizitzaInprimatu();
          System.out.println(" ");
          listaAkzB.akzioaAukeratuEtaBurutu(3);
          //if(!listaAkzB.getMugitu1()){
            Teklatua.getNireTeklatua().emanEnter();
          //}
          listaAkzB.setMugitu(false);
          banketxea.deletePertsonaiaMatrizetik(preX, preY);
          banketxea.setPertsonaiaMatrizean();
          }
          if((listae.etsaiaGuztiakHilda()) || (p.getPv()<=0)){
            listaAkz.pasatuBanketxetikAmaierara();
          }
        }
        if(p.getPv()>0){
          FitxeroakIrakurri.fitxeroaErakutsi("Amaiera.txt");
          Teklatua.getNireTeklatua().emanEnter();
          banketxea.eszenatokiaBukatuOndo();
            Teklatua.getNireTeklatua().emanEnter();
        }
```

```
        else{
          FitxeroakIrakurri.fitxeroaErakutsi("AmaieraTX.txt");
          Teklatua.getNireTeklatua().emanEnter();
            System.out.println("Banketxea mapa errepikatu nahi baduzu,
B idatzi. Bestela, beste edozer idatzi");
            String s = Teklatua.getNireTeklatua().irakurriString();
            String b = "B";
            s.toUpperCase();
            if(s.equals(b)){
              listaAkz.berrizHasieratuBanketxea();

                String izena =
Protagonista.getNireProtagonista().getIzena();
                p=null;
                p=Protagonista.hasieratuProtagonista(izena);
                listae=null;
              banketxea();
            }
            banketxea.eszenatokiaBukatuTxarto();

            Teklatua.getNireTeklatua().emanEnter();
      }
      }
}
```

ListaEtsaiak

```
package packproiektua;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.Random;

public class ListaEtsaiak {

  private ArrayList<Etsaia> lista;

  public ListaEtsaiak(){
    this.lista = new ArrayList<Etsaia>();
  }
  public Iterator<Etsaia> getIteradorea(){
    return this.lista.iterator();
  }

  public int luzera(){
    return this.lista.size();
  }

  public void etsaiaGehitu(Etsaia pEtsaia){
    this.lista.add(pEtsaia);
  }

  public void eraso(){
    Iterator<Etsaia> itr=this.getIteradorea();
    Etsaia e=itr.next();
    e.eraso();
  }

  public Etsaia etsaiaBilatuIzenez(String pIzena) throws
BalioEzEgokia{
```

```
    Iterator<Etsaia> itr=this.getIteradorea();
    Etsaia e=null;
    boolean aurkitua = false;
    String izena=pIzena.toUpperCase();
    while(itr.hasNext() && !aurkitua){
      e=itr.next();
      String etsaiarenIzena=e.getIzena();
      if(etsaiarenIzena.equals(izena)){
        aurkitua = true;
      }
    }
    if(!aurkitua){
      throw new BalioEzEgokia();
    }
    return e;
  }

  public void etsaienBizitzaInprimatu(){
    Iterator<Etsaia> itr=this.getIteradorea();
    Etsaia e=null;
    while(itr.hasNext()){
      e=itr.next();
      if (e.getPv()>0){
        System.out.println(e.getIzena() + " etsaiaren bizitza
"+e.getPv() +" da");
      }
      else{
        System.out.println(e.getIzena() + " etsaia hilda dago");
      }
    }
  }

  public Etsaia returnPosizioan(int pPos) throws NotZenbakiEgokia{
    if(pPos>this.lista.size()){
      throw new NotZenbakiEgokia();
    }
    Iterator<Etsaia>itr=this.getIteradorea();
    int lag=pPos;
    Etsaia e=null;
    while(itr.hasNext()&&lag>0){
      e=itr.next();
      lag--;
    }
    return e;
  }

  public boolean badago(Etsaia pEtsaia) throws BalioEzEgokia{
    Iterator<Etsaia> itr=this.getIteradorea();
    Etsaia e=null;
    boolean aurkitua = false;
    while(itr.hasNext() && !aurkitua){
      e=itr.next();
      if(e.getIzena().equals(pEtsaia.getIzena())){
        aurkitua = true;
      }
    }
    if(!aurkitua){
      throw new BalioEzEgokia();
    }
    return aurkitua;
```

```
}

public void etsaiakInprimatu(){

  Iterator<Etsaia> itr=this.getIteradorea();
  Etsaia e=null;
  while(itr.hasNext()){
    e=itr.next();
    if(!e.hilda()){
      System.out.println(e.getIzena() + " etsaia");
    }
  }
}

public ListaEtsaiak etsaiaHildaBadagoKendu(){
  ListaEtsaiak listaBerria=new ListaEtsaiak();
  Iterator<Etsaia> itr=this.getIteradorea();
  Etsaia e=null;
  while(itr.hasNext()){
    e=itr.next();
    if(!e.hilda()){
      listaBerria.etsaiaGehitu(e);
    }
  }
  return listaBerria;
}

public void etsaiakHilBETATESTER(){

  Iterator<Etsaia> itr=this.getIteradorea();
  Etsaia e=null;
  while(itr.hasNext()){
    e=itr.next();
    e.setBizitza(0);
  }
}

public boolean etsaiaGuztiakHilda(){
  Iterator<Etsaia> itr=this.getIteradorea();
  Etsaia e=null;
  boolean guztiak = true;
  while(itr.hasNext()){
    e=itr.next();
    if(!e.hilda()){
      guztiak = false;
    }
  }
  return guztiak;

}

public void etsaiakSortu(){
  int etskont = 1;
  ListaPertsonaiak pertson=ListaPertsonaiak.nireListaPertsonaiak();
  Banketxea banketxe = Banketxea.getNireBanketxea();
  while (etskont<=7){
    Random rnd=new Random();
    int rx=rnd.nextInt(16)+2;
    int ry=rnd.nextInt(16)+2;
```

```
if(!ListaGordelekuak.getNireListaGordelekuak().okupatutaDago(rx,ry) &&
!pertson.okupatutaDago(rx,ry)){
        if(etskont==1){
          Etsaia eA = new Etsaia("A",rx,ry);
          this.etsaiaGehitu(eA);
          banketxe.setEtsaiakMatrizean(rx, ry, eA.getIzena());
        }
        else if(etskont==2){
          Etsaia eB = new Etsaia("B",rx,ry);
          this.etsaiaGehitu(eB);
          banketxe.setEtsaiakMatrizean(rx, ry, eB.getIzena());
        }
        else if(etskont==3){
          Etsaia eC = new Etsaia("C",rx,ry);
          this.etsaiaGehitu(eC);
          banketxe.setEtsaiakMatrizean(rx, ry, eC.getIzena());
        }
        else if(etskont==4){
          Etsaia eD = new Etsaia("D",rx,ry);
          this.etsaiaGehitu(eD);
          banketxe.setEtsaiakMatrizean(rx, ry, eD.getIzena());
        }
        else if(etskont==5){
          Etsaia eE = new Etsaia("E",rx,ry);
          this.etsaiaGehitu(eE);
          banketxe.setEtsaiakMatrizean(rx, ry, eE.getIzena());
        }
        else if(etskont==6){
          Etsaia eF = new Etsaia("F",rx,ry);
          this.etsaiaGehitu(eF);
          banketxe.setEtsaiakMatrizean(rx, ry, eF.getIzena());
        }
        else if(etskont==7){
          Etsaia eG = new Etsaia("G",rx,ry);
          this.etsaiaGehitu(eG);
          banketxe.setEtsaiakMatrizean(rx, ry, eG.getIzena());
        }
        etskont=etskont+1;
      }

    }
  }
}
```

ListaGordelekuak (EMA)

ListaPertsonaiak (EMA)

Mugitu

```
package packproiektua;
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.Iterator;


public class Mugitu extends Akzioa{

  public Mugitu(){
```

```
    super("Mugitu",4);
  }

  public void mugitu(){
    try{
      ListaAkzioa l = new ListaAkzioa();
      l.setMugitu(true);
      l.setMugitu1(true);
      int lag=noranzkoaLortu();
      Protagonista p=Protagonista.getNireProtagonista();
      Banketxea banketxea = Banketxea.getNireBanketxea();
      int x=p.getX();
      int y=p.getY();
      if(lag==1){
        if(y+1<=18){
          if(y==17) {
            System.out.println("Sartu duzun balioa ez da egokia...");
            p.posizioazAldatu(x,y);
            System.out.println();
            System.out.println("Berriro galdetuko dizut, norantz nahi
duzu mugitu?");
            System.out.println("> W < gorantz joateko");
            System.out.println("> A < ezkerrerantz joateko");
            System.out.println("> S < beherentzat joateko");
            System.out.println("> D < eskuineranz joateko");
            this.mugitu();
          }
          else if((banketxea.matrizekoBalioa(x,
y+1)=='%')||(banketxea.matrizekoBalioa(x,
y+1)=='@')||(banketxea.matrizekoBalioa(x, y+1)=='X')){
            System.out.println("Ezin zara hortik pasa...");
            p.posizioazAldatu(x,y);
            System.out.println();
            System.out.println("Berriro galdetuko dizut, norantz nahi
duzu mugitu?");
            System.out.println("> W < gorantz joateko");
            System.out.println("> A < ezkerrerantz joateko");
            System.out.println("> S < beherentzat joateko");
            System.out.println("> D < eskuineranz joateko");
            this.mugitu();
          }
          else if((banketxea.matrizekoBalioa(x,
y+1)=='A')||(banketxea.matrizekoBalioa(x, y+1)=='B')||
              (banketxea.matrizekoBalioa(x,
y+1)=='C')||(banketxea.matrizekoBalioa(x, y+1)=='D')||
              (banketxea.matrizekoBalioa(x,
y+1)=='E')||(banketxea.matrizekoBalioa(x, y+1)=='F')||
              (banketxea.matrizekoBalioa(x, y+1)=='G')){
            System.out.println("Ezin zara etsai baten gainean
kokatu...");
            p.posizioazAldatu(x,y);
            System.out.println();
            System.out.println("Berriro galdetuko dizut, norantz nahi
duzu mugitu?");
            System.out.println("> W < gorantz joateko");
            System.out.println("> A < ezkerrerantz joateko");
            System.out.println("> S < beherentzat joateko");
            System.out.println("> D < eskuineranz joateko");
            this.mugitu();
          }
          else{
```

```
          p.posizioazAldatu(x,y+1);
          }
        }
      }
      else{
        if(lag==2){
          if(y-1>=1){
            if(y==2) {
              System.out.println("Sartu duzun balioa ez da
egokia...");
              p.posizioazAldatu(x,y);
              System.out.println();
              System.out.println("Berriro galdetuko dizut, norantz
nahi duzu mugitu?");
              System.out.println("> W < gorantz joateko");
              System.out.println("> A < ezkerrerantz joateko");
              System.out.println("> S < beherentzat joateko");
              System.out.println("> D < eskuineranz joateko");
              this.mugitu();
            }
            else if((banketxea.matrizekoBalioa(x,
y+1)=='%')||(banketxea.matrizekoBalioa(x,
y+1)=='@')||(banketxea.matrizekoBalioa(x, y+1)=='X')){
              System.out.println("Ezin zara hortik pasa...");
              p.posizioazAldatu(x,y);
              System.out.println();
              System.out.println("Berriro galdetuko dizut, norantz
nahi duzu mugitu?");
              System.out.println("> W < gorantz joateko");
              System.out.println("> A < ezkerrerantz joateko");
              System.out.println("> S < beherentzat joateko");
              System.out.println("> D < eskuineranz joateko");
              this.mugitu();
            }
            else if((banketxea.matrizekoBalioa(x, y-
1)=='A')||(banketxea.matrizekoBalioa(x, y-1)=='B')||
                (banketxea.matrizekoBalioa(x, y-
1)=='C')||(banketxea.matrizekoBalioa(x, y-1)=='D')||
                (banketxea.matrizekoBalioa(x, y-
1)=='E')||(banketxea.matrizekoBalioa(x, y-1)=='F')||
                (banketxea.matrizekoBalioa(x, y-1)=='G')){
              System.out.println("Ezin zara etsai baten gainean
kokatu...");
              p.posizioazAldatu(x,y);
              System.out.println();
              System.out.println("Berriro galdetuko dizut, norantz
nahi duzu mugitu?");
              System.out.println("> W < gorantz joateko");
              System.out.println("> A < ezkerrerantz joateko");
              System.out.println("> S < beherentzat joateko");
              System.out.println("> D < eskuineranz joateko");
              this.mugitu();
            }
            else{
            p.posizioazAldatu(x,y-1);
            }
          }
        }
        else{
          if(lag==3){
            if(x-1>=1){
```

```
              if(x==2) {
                System.out.println("Sartu duzun balioa ez da
egokia...");
                p.posizioazAldatu(x,y);
                System.out.println();
                System.out.println("Berriro galdetuko dizut, norantz
nahi duzu mugitu?");
                System.out.println("> W < gorantz joateko");
                System.out.println("> A < ezkerrerantz joateko");
                System.out.println("> S < beherentzat joateko");
                System.out.println("> D < eskuineranz joateko");
                this.mugitu();
              }
              else if((banketxea.matrizekoBalioa(x,
y+1)=='%')||(banketxea.matrizekoBalioa(x,
y+1)=='@')||(banketxea.matrizekoBalioa(x, y+1)=='X')){
                System.out.println("Ezin zara hortik pasa...");
                p.posizioazAldatu(x,y);
                System.out.println();
                System.out.println("Berriro galdetuko dizut, norantz
nahi duzu mugitu?");
                System.out.println("> W < gorantz joateko");
                System.out.println("> A < ezkerrerantz joateko");
                System.out.println("> S < beherentzat joateko");
                System.out.println("> D < eskuineranz joateko");
                this.mugitu();
              }
              else if((banketxea.matrizekoBalioa(x-1,
y)=='A')||(banketxea.matrizekoBalioa(x-1, y)=='B')||
                  (banketxea.matrizekoBalioa(x-1,
y)=='C')||(banketxea.matrizekoBalioa(x-1, y)=='D')||
                  (banketxea.matrizekoBalioa(x-1,
y)=='E')||(banketxea.matrizekoBalioa(x-1, y)=='F')||
                  (banketxea.matrizekoBalioa(x-1, y)=='G')){
                System.out.println("Ezin zara etsai baten gainean
kokatu...");
                p.posizioazAldatu(x,y);
                System.out.println();
                System.out.println("Berriro galdetuko dizut, norantz
nahi duzu mugitu?");
                System.out.println("> W < gorantz joateko");
                System.out.println("> A < ezkerrerantz joateko");
                System.out.println("> S < beherentzat joateko");
                System.out.println("> D < eskuineranz joateko");
                this.mugitu();
              }
              else{
              p.posizioazAldatu(x-1,y);
              }
            }
          }
          else{
            if(x+1<=18){
              if(x==17) {
                System.out.println("Sartu duzun balioa ez da
egokia...");
                p.posizioazAldatu(x,y);
                System.out.println();
                System.out.println("Berriro galdetuko dizut, norantz
nahi duzu mugitu?");
                System.out.println("> W < gorantz joateko");
```

```
                System.out.println("> A < ezkerrerantz joateko");
                System.out.println("> S < beherentzat joateko");
                System.out.println("> D < eskuineranz joateko");
                this.mugitu();
              }
              else if((banketxea.matrizekoBalioa(x,
y+1)=='%')||(banketxea.matrizekoBalioa(x,
y+1)=='@')||(banketxea.matrizekoBalioa(x, y+1)=='X')){
                System.out.println("Ezin zara hortik pasa...");
                p.posizioazAldatu(x,y);
                System.out.println();
                System.out.println("Berriro galdetuko dizut, norantz
nahi duzu mugitu?");
                System.out.println("> W < gorantz joateko");
                System.out.println("> A < ezkerrerantz joateko");
                System.out.println("> S < beherentzat joateko");
                System.out.println("> D < eskuineranz joateko");
                this.mugitu();
              }
              else if((banketxea.matrizekoBalioa(x+1,
y)=='A')||(banketxea.matrizekoBalioa(x+1, y)=='B')||
                  (banketxea.matrizekoBalioa(x+1,
y)=='C')||(banketxea.matrizekoBalioa(x+1, y)=='D')||
                  (banketxea.matrizekoBalioa(x+1,
y)=='E')||(banketxea.matrizekoBalioa(x+1, y)=='F')||
                  (banketxea.matrizekoBalioa(x+1, y)=='G')){
                System.out.println("Ezin zara etsai baten gainean
kokatu...");
                p.posizioazAldatu(x,y);
                System.out.println();
                System.out.println("Berriro galdetuko dizut, norantz
nahi duzu mugitu?");
                System.out.println("> W < gorantz joateko");
                System.out.println("> A < ezkerrerantz joateko");
                System.out.println("> S < beherentzat joateko");
                System.out.println("> D < eskuineranz joateko");
                this.mugitu();
              }
              else{
              p.posizioazAldatu(x+1,y);
              }
            }
          }
        }
      }
    }
    catch(BalioEzEgokia lag){
      System.out.println("Sartu duzun balioa ez da egokia...");
    }
  }

  private int noranzkoaLortu() throws BalioEzEgokia{
    char lag=Teklatua.getNireTeklatua().irakurriChar();
    int emaitza=0;
    if((lag=='D')||(lag=='d')){
      emaitza=1;
    }
    else{
      if((lag=='A')||(lag=='a')){
        emaitza=2;
      }
```

```
      else{
        if((lag=='W')||(lag=='w')){
          emaitza=3;
        }
        else{
          if((lag=='S')||(lag=='s')){
            emaitza=4;
          }
          else {
            throw new BalioEzEgokia();
          }
        }
      }
    }
    return emaitza;
  }
}
```

NotZenbakiEgokia

```
package packproiektua;

public class NotZenbakiEgokia extends Exception {

  public NotZenbakiEgokia(){
    super();
  }
}
```

Objetua

```
package packproiektua;

public abstract class Objetua {
  private String izena;

  public Objetua(String pIzena){
    this.izena=pIzena;
  }


  public String getIzena(){
    return this.izena;
  }

  public abstract void objektuaErabili();

}
```

Pertsonaia

```
package packproiektua;

public class Pertsonaia {
  protected int x;
  protected int y;
  protected int pv;
  protected String izena;

  public Pertsonaia(String pIzena){
    this.pv=100;
    this.izena=pIzena;
```

```
  }
  public void pertsonaiaEguneratu(Akzioa pAkzioa){

  }

  public int getX(){
    return this.x;
  }

  public int getY(){
    return this.y;
  }

  public void setKordenatuak(int pX,int pY){
    this.x=pX;
    this.y=pY;
  }
  /*public Pertsonaia hasieratuPertsonaia(){
  }*/

}
```

Pitia

```
package packproiektua;

public class Pitia extends Objetua{

  public Pitia(){
    super("Pitia");
  }

  public void objektuaErabili(){
    Protagonista p=Protagonista.getNireProtagonista();
    p.setPv(p.getPv()+10);

  }

}
```

Protagonista (EMA)

Saloia (EMA)

Teklatua (EMA)

TiroEgin

```
package packproiektua;
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.Iterator;



public class TiroEgin extends Akzioa {

  public TiroEgin(){
    super("Tiro Egin",1);
```

```
  }

  public void tiroEgin() throws FileNotFoundException, IOException{
    Protagonista p = Protagonista.getNireProtagonista();
    int erasoa=Protagonista.getNireProtagonista().getAtaq();
    int defentsa = Protagonista.getNireProtagonista().getDef();
    int bizitza = Protagonista.getNireProtagonista().getPv();
    ListaEtsaiak l1 =
Banketxea.getNireBanketxea().lortuEtsaiakBanketxetik();
    ListaEtsaiak l2 = this.berdinakDira();
    ListaEtsaiak l3=l2.etsaiaHildaBadagoKendu();
    if(l3.luzera()!=0){
      System.out.println("Aukeratu ahal dituzun etsaiak hauek dira:
");
      l2.etsaiakInprimatu();
      System.out.println("Idatzi tirokatu nahi duzun etsaiaren
letra");
      this.tiroketa(l2);
    }
    else if(l3.luzera()==0){
      System.out.println();
      System.out.println("Ezin duzu etsairik tirokatu momentu honetan.
Egizu beste zeozer");
      System.out.println();
        ListaAkzioa listaAkz = new ListaAkzioa();
        ListaAkzioa listaAkzB=listaAkz.listaAkzioakSortu(3);
        listaAkzB.akzioaAukeratuEtaBurutu(3);
    }
  }

  private void tiroketa(ListaEtsaiak pLista){
    boolean kontrolpean;
    do{
      kontrolpean = true;

    try{
      String izena = Teklatua.getNireTeklatua().irakurriString();
      Etsaia e = pLista.etsaiaBilatuIzenez(izena);
      if(pLista.badago(e)){
        //Ataque del protagonista al etsaia
        e.setBizitza(0);
        System.out.println(e.getIzena() + " hilda dago");
      }
      }
      catch(BalioEzEgokia lag){
        System.out.println("Gaizki sartu duzu etsaiaren izena, sartu
berriz...");
        kontrolpean = false;
      }
    }
    while(!kontrolpean);
  }

}
```

AkzioaTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
```

```
import org.junit.Before;
import org.junit.Test;

public class AkzioaTest {
  Akzioa a1,a2,a3;
  ListaEtsaiak ListEts,ListaEts2=new ListaEtsaiak();
  Mugitu m;
  Etsaia e1,e2,e3;

  @Before
  public void setUp() throws Exception {
    a1=new Akzioa("1.Akzioa",1);
    a2=new Akzioa("2.Akzioa",2);
    a3=new Akzioa("3.Akzioa",3);
    e1=new Etsaia("Etsaia1",10,15);
    e2=new Etsaia("Etsaia1",18,10);
    e3=new Etsaia("Etsaia1",14,15);
    Banketxea.getNireBanketxea().eszenatokiaHasieratu();
    ListEts=new ListaEtsaiak();
    ListaEts2=new ListaEtsaiak();
    ListEts.etsaiaGehitu(e1);
    ListEts.etsaiaGehitu(e2);
    ListEts.etsaiaGehitu(e3);
    Protagonista.hasieratuProtagonista("Proba");
    Protagonista.getNireProtagonista().hasierakoPosizioa(3);
  }

  @After
  public void tearDown() throws Exception {
    ListEts.erreseteatu();
  }

  @Test
  public void testAkzioa() {
    assertNotNull(a1);
    assertNotNull(a2);
    assertNotNull(a3);
  }

  @Test
  public void testKutxa() {

    assertFalse(a1.getKutxa());
    assertFalse(a2.getKutxa());
    assertFalse(a3.getKutxa());

    assertTrue(a1.kutxa());
    assertTrue(a2.kutxa());
    assertTrue(a3.kutxa());
  }

  @Test
  public void testBerdinakDira() {
    //ezin da egin (Memorian azalduta)
  }
}
```

DadoaTest

```
package packproiektua;
```

```
import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class DadoaTest {
  Dadoa d=new Dadoa(6);

  @Before
  public void setUp() throws Exception {
    //Dadoa d=new Dadoa(6);
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testDadoa() {
    assertNotNull(d);
  }

  @Test
  public void testBota() {
    d.bota();
    assertTrue(d.getGoikoAldea()<=6);
    assertTrue(d.getGoikoAldea()>=0);
  }

}
```

EtsaiaTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class EtsaiaTest {
  Etsaia e1,e2,e3;

  @Before
  public void setUp() throws Exception {
    e1=new Etsaia("Lehenengo Etsaia",10,10);
    e2=new Etsaia("Bigarren Etsaia",11,11);
    e3=new Etsaia("Hirugarren Etsaia",12,12);
    Protagonista.hasieratuProtagonista("Protagonista proba");
  }

  @After
  public void tearDown() throws Exception {
  }


  @Test
  public void testEtsaia() {
    assertNotNull(e1);
```

```
    assertNotNull(e2);
    assertNotNull(e3);
  }

  @Test
  public void testEraso() {

    e1.eraso();
    if(Protagonista.getNireProtagonista().estalita()){
      assertEquals(Protagonista.getNireProtagonista().getPv(),250);
    }
    else {
      assertTrue(Protagonista.getNireProtagonista().getPv()<=240);
    }
  }

  @Test
  public void testHilda() {

    assertFalse(e1.hilda());
    e1.setBizitza(0);
    assertTrue(e1.hilda());

    assertFalse(e2.hilda());
    e2.setBizitza(0);
    assertTrue(e2.hilda());

    assertFalse(e3.hilda());
    e3.setBizitza(0);
    assertTrue(e3.hilda());
  }
}
```

GordelekuakTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class GordelekuakTest {
  Gordelekua g1,g2,g3,g4;

  @Before
  public void setUp() throws Exception {
    g1=new Gordelekua(0,0);
    g2=new Gordelekua(0,1);
    g3=new Gordelekua(1,0);
    g4=new Gordelekua(1,1);
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testGordelekua() {
    assertNotNull(g1);
```

```
    assertNotNull(g2);
    assertNotNull(g3);
    assertNotNull(g4);
  }

  @Test
  public void testGetX() {
    assertEquals(g1.getX(),0);
    assertEquals(g2.getX(),0);
    assertEquals(g3.getX(),1);
    assertEquals(g4.getX(),1);
  }

  @Test
  public void testGetY() {
    assertEquals(g1.getY(),0);
    assertEquals(g2.getY(),1);
    assertEquals(g3.getY(),0);
    assertEquals(g4.getY(),1);
  }

}
```

HilerriaTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class HilerriaTest {
  ListaAkzioa l;

  @Before
  public void setUp() throws Exception {
    Hilerria.getNireHilerria().eszenatokiaHasieratu();
    Protagonista.hasieratuProtagonista("Protagonista proba");
    l=new ListaAkzioa();
    l.listaAkzioakSortu(2);
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testGetNireHilerria() {
    assertNotNull(Hilerria.getNireHilerria());
  }

  @Test
  public void testMatrizekoBalioa() {
    assertEquals(Hilerria.getNireHilerria().matrizekoBalioa(10,10),
'|');
    assertEquals(Hilerria.getNireHilerria().matrizekoBalioa(1,1),
'+');
  }
```

```
}
```

InbentarioaTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class InbentarioaTest {
  Inbentarioa inb;
  Objetua oP,oK,oL;
  Protagonista p;

  @Before
  public void setUp() throws Exception {
    Inbentarioa.getNireInbentarioa();
    Protagonista.hasieratuProtagonista("Protagonista proba");
    Inbentarioa.getNireInbentarioa().inbentarioaSortu();
  }

  @After
  public void tearDown() throws Exception{
  }

  @Test
  public void testGetNireInbentarioa(){
    assertNotNull(Inbentarioa.getNireInbentarioa());
  }

  @Test
  public void testInbentarioaSortu() {

assertNotNull(Inbentarioa.getNireInbentarioa().inbentarioaSortu());
  }

  @Test
  public void testObjetuaErabili() {
    //Pitia erabiltzen du
    Inbentarioa.getNireInbentarioa().objetuaErabili("Pitia");
    assertEquals(Protagonista.getNireProtagonista().getPv(),250,5);

    //Likorea erabiltzen badu
    Inbentarioa.getNireInbentarioa().objetuaErabili("Likorea");
    assertEquals(Protagonista.getNireProtagonista().getPv(),350,5);

    //Kapela erabiltzen badu
    Inbentarioa.getNireInbentarioa().objetuaErabili("Kapela");
    assertEquals(Protagonista.getNireProtagonista().getPv(),350,5);
  }

}
```

KapelaTest

```
package packproiektua;

import static org.junit.Assert.*;
```

```
import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class KapelaTest {

  @Before
  public void setUp() throws Exception {
    Protagonista.hasieratuProtagonista("Proba");
    Inbentarioa.getNireInbentarioa().inbentarioaSortu();
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testObjektuaErabili() {
    Protagonista.getNireProtagonista().objetuaErabili("Kapela");
    assertEquals(Protagonista.getNireProtagonista().getPv(),300);
  }

}
```

LikoreaTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class LikoreaTest {

  @Before
  public void setUp() throws Exception {
    Protagonista.hasieratuProtagonista("Proba");
    Inbentarioa.getNireInbentarioa().inbentarioaSortu();
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testObjektuaErabili() {
    Protagonista.getNireProtagonista().objetuaErabili("Likorea");
    assertEquals(Protagonista.getNireProtagonista().getPv(),350);
  }
}
```

ListaAkzioaTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
```

```
import org.junit.Before;
import org.junit.Test;

public class ListaAkzioaTest {
  ListaAkzioa ls,lh,lb,ls1,lh1,lb1,ls2,lh2,lb2;
  Akzioa a;

  @Before
  public void setUp() throws Exception {
    ls=new ListaAkzioa();
    ls1=new ListaAkzioa();
    lh=new ListaAkzioa();
    lh1=new ListaAkzioa();
    lb=new ListaAkzioa();
    lb1=new ListaAkzioa();
    ls2=new ListaAkzioa();
    lh2=new ListaAkzioa();
    lb2=new ListaAkzioa();
    ls1=ls.listaAkzioakSortu(1);
    lh1=lh.listaAkzioakSortu(2);
    lb1=lb.listaAkzioakSortu(3);
  }

  @After
  public void tearDown() throws Exception {
    ls1.clear();
    lh1.clear();
    lb1.clear();
  }

  @Test
  public void testGetEliza() {

    assertFalse(lh1.getEliza());
    lh1.setElizaT();
    assertTrue(lh1.getEliza());

  }

  @Test
  public void testListaAkzioakSortu() {
    assertTrue(ls2.hutsa());
    ls2=ls.listaAkzioakSortu(1);
    assertFalse(ls1.hutsa());

    assertTrue(lh2.hutsa());
    lh2=lh.listaAkzioakSortu(2);
    assertFalse(lh2.hutsa());

    assertTrue(lb2.hutsa());
    lb2=lb.listaAkzioakSortu(3);
    assertFalse(lb2.hutsa());

  }

}
```

ListaEgoerakTest

```
package packproiektua;
```

```
import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class ListaEgoerakTest {

  @Before
  public void setUp() throws Exception {
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testGetNireListaEgoerak() {
    fail("Not yet implemented");
  }

  @Test
  public void testEgoeraEguneratu() {
    fail("Not yet implemented");
  }

  @Test
  public void testHasieratuEgoerak() {
    fail("Not yet implemented");
  }

  @Test
  public void testMain() {
    fail("Not yet implemented");
  }

  @Test
  public void testBanketxea() {
    fail("Not yet implemented");
  }

}
```

ListaEtsaiakTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class ListaEtsaiakTest {
  ListaEtsaiak l1;
  Etsaia e1,e2;

  @Before
  public void setUp() throws Exception {
    l1=new ListaEtsaiak();
    l1.etsaiakSortu();
```

```
  e1=new Etsaia("Etsaia",15,16);
  l1.etsaiaGehitu(e1);
  e2=new Etsaia("Etsaia2",16,19);
  Protagonista.hasieratuProtagonista("Protagonista_Proba");
}

@After
public void tearDown() throws Exception {
  l1.erreseteatu();
}

@Test
public void testListaEtsaiak() {
  assertNotNull(l1);
}


@Test
public void testLuzera() {
  assertEquals(l1.luzera(),8);
}

@Test
public void testEtsaiaGehitu() {
  l1.etsaiaGehitu(e2);
  assertEquals(l1.luzera(),9);
}

@Test
public void testEraso() {
  l1.eraso();
  assertTrue(Protagonista.getNireProtagonista().getPv()<=250);
}

@Test
public void testEtsaiaBilatuIzenez() {
  try{
  assertEquals(e1, l1.etsaiaBilatuIzenez("Etsaia"));
  l1.etsaiaGehitu(e2);
  assertEquals(e2,l1.etsaiaBilatuIzenez("Etsaia2"));
  }
  catch(BalioEzEgokia e){
    System.out.println("Ez da aurkitu etsaia hori");
  }
}


@Test
public void testReturnPosizioan() throws NotZenbakiEgokia {
  assertEquals(l1.returnPosizioan(8),e1);
}

@Test
public void testBadago() throws BalioEzEgokia {
  assertTrue(l1.badago(e1));
  l1.etsaiaGehitu(e2);
  assertTrue(l1.badago(e2));
}

@Test
public void testEtsaiaHildaBadagoKendu() {
```

```
    e1.setBizitza(0);
    l1.etsaiaHildaBadagoKendu();
    assertEquals(l1.luzera(),8);
  }

  @Test
  public void testEtsaiakHilBETATESTER() {
    l1.etsaiakHilBETATESTER();
    assertTrue(l1.etsaiaGuztiakHilda());
  }

  @Test
  public void testEtsaiaGuztiakHilda() {
    l1.etsaiakHilBETATESTER();
    l1.etsaiaHildaBadagoKendu();
    assertTrue(l1.etsaiaGuztiakHilda());
  }

  @Test
  public void testEtsaiakSortu() {
    assertNotNull(l1);
  }
}
```

ListaGordelekuakTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class ListaGordelekuakTest {
  ListaGordelekuak l;

  @Before
  public void setUp() throws Exception {
    l=ListaGordelekuak.getNireListaGordelekuak();
    l.gordelekuakSortu();
  }

  @After
  public void tearDown() throws Exception {
    l.erreseteatu();
  }

  @Test
  public void testGetNireListaGordelekuak() {
    assertNotNull(l);
  }

  @Test
  public void testOkupatutaDago() {
    assertFalse(l.okupatutaDago(4,4));
    assertTrue(l.okupatutaDago(5,5));
    assertTrue(l.okupatutaDago(5,6));
    assertTrue(l.okupatutaDago(6,5));
    assertTrue(l.okupatutaDago(6,6));
    assertTrue(l.okupatutaDago(15,15));
```

```
    assertTrue(l.okupatutaDago(15,16));
    assertTrue(l.okupatutaDago(16,15));
    assertTrue(l.okupatutaDago(16,16));
  }


  @Test
  public void testGordelekuakSortu() {
    l.erreseteatu();
    assertEquals(l.luzera(),0);
    l.gordelekuakSortu();
    assertEquals(l.luzera(),8);

  }

}
```

ListaPertsonaiakTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class ListaPertsonaiakTest {
  Pertsonaia p1,p2,p3;

  @Before
  public void setUp() throws Exception {
    p1=new Pertsonaia("Pertsonaia1");
    p2=new Pertsonaia("Pertsonaia2");
    p3=new Pertsonaia("Pertsonaia1");
    ListaPertsonaiak.nireListaPertsonaiak().gehituPertsonaia(p1);
    ListaPertsonaiak.nireListaPertsonaiak().gehituPertsonaia(p2);
    ListaPertsonaiak.nireListaPertsonaiak().gehituPertsonaia(p3);
    p1.setKordenatuak(14,14);
    p2.setKordenatuak(16,17);
    p3.setKordenatuak(18,16);
  }

  @After
  public void tearDown() throws Exception {
    ListaPertsonaiak.nireListaPertsonaiak().erreseteatu();
  }

  @Test
  public void testNireListaPertsonaiak() {
    assertNotNull(ListaPertsonaiak.nireListaPertsonaiak());
  }

  @Test
  public void testPertsonaiaBerdina() {

assertTrue(ListaPertsonaiak.nireListaPertsonaiak().pertsonaiaBerdina(p
1));
  }

  @Test
```

```
  public void testOkupatutaDago() {

assertFalse(ListaPertsonaiak.nireListaPertsonaiak().okupatutaDago(15,1
5));

assertTrue(ListaPertsonaiak.nireListaPertsonaiak().okupatutaDago(14,14
));

assertTrue(ListaPertsonaiak.nireListaPertsonaiak().okupatutaDago(16,17
));

assertTrue(ListaPertsonaiak.nireListaPertsonaiak().okupatutaDago(18,16
));
  }

}
```

MugituTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class MugituTest {

  Mugitu m1,m2,m3;

  @Before
  public void setUp() throws Exception {
    m1=new Mugitu();
    m2=new Mugitu();
    m3=new Mugitu();
    Protagonista.hasieratuProtagonista("Proba");
    Protagonista.getNireProtagonista().hasierakoPosizioa(3); //Akzio
hau bakarrik banketxean dago
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testMugitu() {
    assertNotNull(m1);
    assertNotNull(m2);
    assertNotNull(m3);
  }

  @Test
  public void testMugitu1() {

    //goruntz badoa

    ListaAkzioa l = new ListaAkzioa();
    l.setMugitu(true);
    l.setMugitu1(true);
    int lag=1;  //Gora mugitu
```

```
    Protagonista p=Protagonista.getNireProtagonista();
    Banketxea banketxea = Banketxea.getNireBanketxea();
    int x=p.getX();
    int y=p.getY();
    if(lag==1){
      if(y+1<=18){
        if(y==17) {
          p.posizioazAldatu(x,y);
        }
        else if((banketxea.matrizekoBalioa(x,
y+1)=='%')||(banketxea.matrizekoBalioa(x, y+1)=='@')){
          p.posizioazAldatu(x,y);
        }
        else if((banketxea.matrizekoBalioa(x,
y+1)=='A')||(banketxea.matrizekoBalioa(x, y+1)=='B')||
            (banketxea.matrizekoBalioa(x,
y+1)=='C')||(banketxea.matrizekoBalioa(x, y+1)=='D')||
            (banketxea.matrizekoBalioa(x,
y+1)=='E')||(banketxea.matrizekoBalioa(x, y+1)=='F')||
            (banketxea.matrizekoBalioa(x, y+1)=='G')){
          p.posizioazAldatu(x,y);
        }
        else{
        p.posizioazAldatu(x,y+1);
        }
      }
    }
    else{
      if(lag==2){
        if(y-1>=1){
          if(y==2) {
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x, y-
1)=='%')||(banketxea.matrizekoBalioa(x, y-1)=='@')){
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x, y-
1)=='A')||(banketxea.matrizekoBalioa(x, y-1)=='B')||
              (banketxea.matrizekoBalioa(x, y-
1)=='C')||(banketxea.matrizekoBalioa(x, y-1)=='D')||
              (banketxea.matrizekoBalioa(x, y-
1)=='E')||(banketxea.matrizekoBalioa(x, y-1)=='F')||
              (banketxea.matrizekoBalioa(x, y-1)=='G')){
            p.posizioazAldatu(x,y);
          }
          else{
          p.posizioazAldatu(x,y-1);
          }
        }
      }
      else{
        if(lag==3){
          if(x-1>=1){
            if(x==2) {
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x-1,
y)=='%')||(banketxea.matrizekoBalioa(x-1, y)=='@')){
              p.posizioazAldatu(x,y);
            }
```

```
            else if((banketxea.matrizekoBalioa(x-1,
y)=='A')||(banketxea.matrizekoBalioa(x-1, y)=='B')||
                  (banketxea.matrizekoBalioa(x-1,
y)=='C')||(banketxea.matrizekoBalioa(x-1, y)=='D')||
                  (banketxea.matrizekoBalioa(x-1,
y)=='E')||(banketxea.matrizekoBalioa(x-1, y)=='F')||
                  (banketxea.matrizekoBalioa(x-1, y)=='G')){
              p.posizioazAldatu(x,y);
            }
            else{
            p.posizioazAldatu(x-1,y);
            }
          }
        }
        else{
          if(x+1<=18){
            if(x==17) {
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x+1,
y)=='%')||(banketxea.matrizekoBalioa(x+1, y)=='@')){
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x+1,
y)=='A')||(banketxea.matrizekoBalioa(x+1, y)=='B')||
                  (banketxea.matrizekoBalioa(x+1,
y)=='C')||(banketxea.matrizekoBalioa(x+1, y)=='D')||
                  (banketxea.matrizekoBalioa(x+1,
y)=='E')||(banketxea.matrizekoBalioa(x+1, y)=='F')||
                  (banketxea.matrizekoBalioa(x+1, y)=='G')){
              p.posizioazAldatu(x,y);
            }
            else{
            p.posizioazAldatu(x+1,y);
            }
          }
        }
      }
    }

    assertEquals(Protagonista.getNireProtagonista().getY(),9);

    //Beheruntz badoa

    Protagonista.hasieratuProtagonista("Proba");
    lag=2;

    if(lag==1){
      if(y+1<=18){
        if(y==17) {
          p.posizioazAldatu(x,y);
        }
        else if((banketxea.matrizekoBalioa(x,
y+1)=='%')||(banketxea.matrizekoBalioa(x, y+1)=='@')){
          p.posizioazAldatu(x,y);
        }
        else if((banketxea.matrizekoBalioa(x,
y+1)=='A')||(banketxea.matrizekoBalioa(x, y+1)=='B')||
              (banketxea.matrizekoBalioa(x,
y+1)=='C')||(banketxea.matrizekoBalioa(x, y+1)=='D')||
```

```
            (banketxea.matrizekoBalioa(x,
y+1)=='E')||(banketxea.matrizekoBalioa(x, y+1)=='F')||
            (banketxea.matrizekoBalioa(x, y+1)=='G')){
          p.posizioazAldatu(x,y);
        }
        else{
        p.posizioazAldatu(x,y+1);
        }
      }
    }
    else{
      if(lag==2){
        if(y-1>=1){
          if(y==2) {
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x, y-
1)=='%')||(banketxea.matrizekoBalioa(x, y-1)=='@')){
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x, y-
1)=='A')||(banketxea.matrizekoBalioa(x, y-1)=='B')||
              (banketxea.matrizekoBalioa(x, y-
1)=='C')||(banketxea.matrizekoBalioa(x, y-1)=='D')||
              (banketxea.matrizekoBalioa(x, y-
1)=='E')||(banketxea.matrizekoBalioa(x, y-1)=='F')||
              (banketxea.matrizekoBalioa(x, y-1)=='G')){
            p.posizioazAldatu(x,y);
          }
          else{
          p.posizioazAldatu(x,y-1);
          }
        }
      }
      else{
        if(lag==3){
          if(x-1>=1){
            if(x==2) {
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x-1,
y)=='%')||(banketxea.matrizekoBalioa(x-1, y)=='@')){
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x-1,
y)=='A')||(banketxea.matrizekoBalioa(x-1, y)=='B')||
                (banketxea.matrizekoBalioa(x-1,
y)=='C')||(banketxea.matrizekoBalioa(x-1, y)=='D')||
                (banketxea.matrizekoBalioa(x-1,
y)=='E')||(banketxea.matrizekoBalioa(x-1, y)=='F')||
                (banketxea.matrizekoBalioa(x-1, y)=='G')){
              p.posizioazAldatu(x,y);
            }
            else{
            p.posizioazAldatu(x-1,y);
            }
          }
        }
        else{
          if(x+1<=18){
            if(x==17) {
```

```
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x+1,
y)=='%')||(banketxea.matrizekoBalioa(x+1, y)=='@')){
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x+1,
y)=='A')||(banketxea.matrizekoBalioa(x+1, y)=='B')||
              (banketxea.matrizekoBalioa(x+1,
y)=='C')||(banketxea.matrizekoBalioa(x+1, y)=='D')||
              (banketxea.matrizekoBalioa(x+1,
y)=='E')||(banketxea.matrizekoBalioa(x+1, y)=='F')||
              (banketxea.matrizekoBalioa(x+1, y)=='G')){
            p.posizioazAldatu(x,y);
          }
          else{
          p.posizioazAldatu(x+1,y);
          }
        }
      }
    }

    assertEquals(Protagonista.getNireProtagonista().getY(),7);

    //Ezkerrera badoa

    Protagonista.hasieratuProtagonista("Proba");
    lag=3;

    if(lag==1){
      if(y+1<=18){
        if(y==17) {
          p.posizioazAldatu(x,y);
        }
        else if((banketxea.matrizekoBalioa(x,
y+1)=='%')||(banketxea.matrizekoBalioa(x, y+1)=='@')){
          p.posizioazAldatu(x,y);
        }
        else if((banketxea.matrizekoBalioa(x,
y+1)=='A')||(banketxea.matrizekoBalioa(x, y+1)=='B')||
            (banketxea.matrizekoBalioa(x,
y+1)=='C')||(banketxea.matrizekoBalioa(x, y+1)=='D')||
            (banketxea.matrizekoBalioa(x,
y+1)=='E')||(banketxea.matrizekoBalioa(x, y+1)=='F')||
            (banketxea.matrizekoBalioa(x, y+1)=='G')){
          p.posizioazAldatu(x,y);
        }
        else{
        p.posizioazAldatu(x,y+1);
        }
      }
    }
    else{
      if(lag==2){
        if(y-1>=1){
          if(y==2) {
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x, y-
1)=='%')||(banketxea.matrizekoBalioa(x, y-1)=='@')){
```

```
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x, y-
1)=='A')||(banketxea.matrizekoBalioa(x, y-1)=='B')||
              (banketxea.matrizekoBalioa(x, y-
1)=='C')||(banketxea.matrizekoBalioa(x, y-1)=='D')||
              (banketxea.matrizekoBalioa(x, y-
1)=='E')||(banketxea.matrizekoBalioa(x, y-1)=='F')||
              (banketxea.matrizekoBalioa(x, y-1)=='G')){
            p.posizioazAldatu(x,y);
          }
          else{
          p.posizioazAldatu(x,y-1);
          }
        }
      }
      else{
        if(lag==3){
          if(x-1>=1){
            if(x==2) {
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x-1,
y)=='%')||(banketxea.matrizekoBalioa(x-1, y)=='@')){
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x-1,
y)=='A')||(banketxea.matrizekoBalioa(x-1, y)=='B')||
                (banketxea.matrizekoBalioa(x-1,
y)=='C')||(banketxea.matrizekoBalioa(x-1, y)=='D')||
                (banketxea.matrizekoBalioa(x-1,
y)=='E')||(banketxea.matrizekoBalioa(x-1, y)=='F')||
                (banketxea.matrizekoBalioa(x-1, y)=='G')){
              p.posizioazAldatu(x,y);
            }
            else{
            p.posizioazAldatu(x-1,y);
            }
          }
        }
        else{
          if(x+1<=18){
            if(x==17) {
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x+1,
y)=='%')||(banketxea.matrizekoBalioa(x+1, y)=='@')){
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x+1,
y)=='A')||(banketxea.matrizekoBalioa(x+1, y)=='B')||
                (banketxea.matrizekoBalioa(x+1,
y)=='C')||(banketxea.matrizekoBalioa(x+1, y)=='D')||
                (banketxea.matrizekoBalioa(x+1,
y)=='E')||(banketxea.matrizekoBalioa(x+1, y)=='F')||
                (banketxea.matrizekoBalioa(x+1, y)=='G')){
              p.posizioazAldatu(x,y);
            }
            else{
            p.posizioazAldatu(x+1,y);
            }
```

```
            }
          }
        }
      }

      assertEquals(Protagonista.getNireProtagonista().getX(),17);

      //Eskumara badoa

      Protagonista.hasieratuProtagonista("Proba");
      lag=4;

      if(lag==1){
        if(y+1<=18){
          if(y==17) {
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x,
y+1)=='%')||(banketxea.matrizekoBalioa(x, y+1)=='@')){
            p.posizioazAldatu(x,y);
          }
          else if((banketxea.matrizekoBalioa(x,
y+1)=='A')||(banketxea.matrizekoBalioa(x, y+1)=='B')||
                (banketxea.matrizekoBalioa(x,
y+1)=='C')||(banketxea.matrizekoBalioa(x, y+1)=='D')||
                (banketxea.matrizekoBalioa(x,
y+1)=='E')||(banketxea.matrizekoBalioa(x, y+1)=='F')||
                (banketxea.matrizekoBalioa(x, y+1)=='G')){
            p.posizioazAldatu(x,y);
          }
          else{
          p.posizioazAldatu(x,y+1);
          }
        }
      }
      else{
        if(lag==2){
          if(y-1>=1){
            if(y==2) {
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x, y-
1)=='%')||(banketxea.matrizekoBalioa(x, y-1)=='@')){
              p.posizioazAldatu(x,y);
            }
            else if((banketxea.matrizekoBalioa(x, y-
1)=='A')||(banketxea.matrizekoBalioa(x, y-1)=='B')||
                  (banketxea.matrizekoBalioa(x, y-
1)=='C')||(banketxea.matrizekoBalioa(x, y-1)=='D')||
                  (banketxea.matrizekoBalioa(x, y-
1)=='E')||(banketxea.matrizekoBalioa(x, y-1)=='F')||
                  (banketxea.matrizekoBalioa(x, y-1)=='G')){
              p.posizioazAldatu(x,y);
            }
            else{
            p.posizioazAldatu(x,y-1);
            }
          }
        }
        else{
          if(lag==3){
```

```
            if(x-1>=1){
              if(x==2) {
                p.posizioazAldatu(x,y);
              }
              else if((banketxea.matrizekoBalioa(x-1,
y)=='%')||(banketxea.matrizekoBalioa(x-1, y)=='@')){
                p.posizioazAldatu(x,y);
              }
              else if((banketxea.matrizekoBalioa(x-1,
y)=='A')||(banketxea.matrizekoBalioa(x-1, y)=='B')||
                    (banketxea.matrizekoBalioa(x-1,
y)=='C')||(banketxea.matrizekoBalioa(x-1, y)=='D')||
                    (banketxea.matrizekoBalioa(x-1,
y)=='E')||(banketxea.matrizekoBalioa(x-1, y)=='F')||
                    (banketxea.matrizekoBalioa(x-1, y)=='G')){
                p.posizioazAldatu(x,y);
              }
              else{
              p.posizioazAldatu(x-1,y);
              }
            }
          }
          else{
            if(x+1<=18){
              if(x==17) {
                p.posizioazAldatu(x,y);
              }
              else if((banketxea.matrizekoBalioa(x+1,
y)=='%')||(banketxea.matrizekoBalioa(x+1, y)=='@')){
                p.posizioazAldatu(x,y);
              }
              else if((banketxea.matrizekoBalioa(x+1,
y)=='A')||(banketxea.matrizekoBalioa(x+1, y)=='B')||
                    (banketxea.matrizekoBalioa(x+1,
y)=='C')||(banketxea.matrizekoBalioa(x+1, y)=='D')||
                    (banketxea.matrizekoBalioa(x+1,
y)=='E')||(banketxea.matrizekoBalioa(x+1, y)=='F')||
                    (banketxea.matrizekoBalioa(x+1, y)=='G')){
                p.posizioazAldatu(x,y);
              }
              else{
              p.posizioazAldatu(x+1,y);
              }
            }
          }
        }
      }

    assertEquals(Protagonista.getNireProtagonista().getX(),17);
  }

}
```

PertsonaiaTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;
```

```
public class PertsonaiaTest {
  Pertsonaia p1,p2,p3;

  @Before
  public void setUp() throws Exception {
    p1=new Pertsonaia("Pertsonaia1");
    p2=new Pertsonaia("Pertsonaia2");
    p3=new Pertsonaia("Pertsonaia3");
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testPertsonaia() {
    assertNotNull(p1);
    assertNotNull(p2);
    assertNotNull(p3);
  }
}
```

ProtagonistaTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class ProtagonistaTest {
  ;

  @Before
  public void setUp() throws Exception {
    Protagonista.hasieratuProtagonista("Proba");
    Inbentarioa.getNireInbentarioa().inbentarioaSortu();
    ListaGordelekuak.getNireListaGordelekuak().gordelekuakSortu();
  }

  @After
  public void tearDown() throws Exception {

  }

  @Test
  public void testGetNireProtagonista() {
    assertNotNull(Protagonista.getNireProtagonista());
  }

  @Test
  public void testObjetuaErabili() {

    //pitia erabili
    Protagonista.getNireProtagonista().objetuaErabili("Pitia");
    assertEquals(Protagonista.getNireProtagonista().getPv(),250);

    //kapela erabili
```

```
    Protagonista.getNireProtagonista().objetuaErabili("Kapela");
    assertEquals(Protagonista.getNireProtagonista().getPv(),250);
//este deberia de dar bien con 300

    //pitia erabili
    Protagonista.getNireProtagonista().objetuaErabili("Likorea");
    assertEquals(Protagonista.getNireProtagonista().getPv(),350);
  }

  @Test
  public void testHasierakoPosizioa() {

    //Lehenengo Egoeran (Saloia)
    Protagonista.getNireProtagonista().hasierakoPosizioa(1);
    assertEquals(Protagonista.getNireProtagonista().getX(),18);
    assertEquals(Protagonista.getNireProtagonista().getY(),15);

    //Bigarren egoeran (hilerria)
    Protagonista.getNireProtagonista().hasierakoPosizioa(2);
    assertEquals(Protagonista.getNireProtagonista().getX(),14);
    assertEquals(Protagonista.getNireProtagonista().getY(),1);

    //Hirugarren egoeran (banketxea)
    Protagonista.getNireProtagonista().hasierakoPosizioa(3);
    assertEquals(Protagonista.getNireProtagonista().getX(),18);
    assertEquals(Protagonista.getNireProtagonista().getY(),8);
  }

  @Test
  public void testEstalita() {

    Protagonista.getNireProtagonista().posizioazAldatu(5,5);
    assertTrue(Protagonista.getNireProtagonista().estalita());

    Protagonista.getNireProtagonista().posizioazAldatu(5,6);
    assertTrue(Protagonista.getNireProtagonista().estalita());

    Protagonista.getNireProtagonista().posizioazAldatu(6,5);
    assertTrue(Protagonista.getNireProtagonista().estalita());

    Protagonista.getNireProtagonista().posizioazAldatu(6,6);
    assertTrue(Protagonista.getNireProtagonista().estalita());

    Protagonista.getNireProtagonista().posizioazAldatu(15,15);
    assertTrue(Protagonista.getNireProtagonista().estalita());

    Protagonista.getNireProtagonista().posizioazAldatu(15,16);
    assertTrue(Protagonista.getNireProtagonista().estalita());

    Protagonista.getNireProtagonista().posizioazAldatu(16,15);
    assertTrue(Protagonista.getNireProtagonista().estalita());

    Protagonista.getNireProtagonista().posizioazAldatu(16,16);
    assertTrue(Protagonista.getNireProtagonista().estalita());
  }

}
```

SaloiaTest

```
package packproiektua;
```

```
import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class SaloiaTest {
  ListaAkzioa l;

  @Before
  public void setUp() throws Exception {
    Saloia.getNireSaloia().eszenatokiaHasieratu();
    Protagonista.hasieratuProtagonista("Protagonista proba");
    l=new ListaAkzioa();
    l.listaAkzioakSortu(2);
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testGetNireSaloia() {
    assertNotNull(Saloia.getNireSaloia());
  }

  @Test
  public void testMatrizekoBalioa() {
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(10,10), ' ');
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(1,1), '+');
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(7,8), ' ');
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(4,9), ' ');
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(8,14), ' ');
  }


  @Test
  public void testIsNullOrEmpty() {
    assertTrue(Saloia.getNireSaloia().isNullOrEmpty(""));
    assertFalse(Saloia.getNireSaloia().isNullOrEmpty("Kaixo"));
  }

}
```

TiroEginTest

```
package packproiektua;

import static org.junit.Assert.*;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class SaloiaTest {
  ListaAkzioa l;

  @Before
  public void setUp() throws Exception {
    Saloia.getNireSaloia().eszenatokiaHasieratu();
    Protagonista.hasieratuProtagonista("Protagonista proba");
```

```
    l=new ListaAkzioa();
    l.listaAkzioakSortu(2);
  }

  @After
  public void tearDown() throws Exception {
  }

  @Test
  public void testGetNireSaloia() {
    assertNotNull(Saloia.getNireSaloia());
  }

  @Test
  public void testMatrizekoBalioa() {
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(10,10), ' ');
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(1,1), '+');
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(7,8), ' ');
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(4,9), ' ');
    assertEquals(Saloia.getNireSaloia().matrizekoBalioa(8,14), ' ');
  }


  @Test
  public void testIsNullOrEmpty() {
    assertTrue(Saloia.getNireSaloia().isNullOrEmpty(""));
    assertFalse(Saloia.getNireSaloia().isNullOrEmpty("Kaixo"));
  }

}
```