Examen réparti 1 – Street Fighter II ^{TM} ! Une solution

Binh-Minh Bui-Xuan

12 mars 2014

Une solution à l'énoncé de l'examen réparti 1 pour le cours CPS 2014.

Personnage

service : Personnage
types : String, int, boolean
observators :
  const nom : [Personnage] → String
  const largeur : [Personnage] → int
  const hauteur : [Personnage] → int
  const force : [Personnage] → int
  pointsDeVie : [Personnage] → int
  estVaincu : [Personnage] → boolean
Constructors :
  init : String × int × int × int × int → [Personnage]
    pre init(nom,largeur,hauteur,force,pointsVie) **require** nom ≠ "" ∧ largeur%2=1 ∧ hauteur%2=1 ∧ 0 < force < pointsVie
Operators :
  retrait : [Personnage] × int → [Personnage]
    pre retrait(P,s) **require** ¬estVaincu(P) ∧ s>0
Observations :
[invariants]
  estVaincu(P) $\stackrel{\min}{=}$ pointsDeVie(P) ≤ 0
[init]
  nom(init(n,l,h,f,p))=n
  largeur(init(n,l,h,f,p))=l
  hauteur(init(n,l,h,f,p))=h
  force(init(n,l,h,f,p))=f
  pointsDeVie(init(n,l,h,f,p))=p
[retrait]
  pointsDeVie(retrait(P,s))=pointsDeVie(P) -s

MoteurJeu

service : MoteurJeu
use : GestionCombat
types : boolean, int, enum RESULTAT{GUILEGAGNANT, GUILEPERDANT, NULLE},
enum COMMANDE{RIEN, GAUCHE, DROITE, FRAPPE}
observators :
const maxPasJeu : [MoteurJeu] → int
pasJeuCourant : [MoteurJeu] → int
estFini : [MoteurJeu] → boolean
resultatFinal : [MoteurJeu] → RESULTAT
pre resultatFinal(M) **require** estFini(M)
combat : [MoteurJeu] → GestionCombat
Constructors :
init : int × int × int → [MoteurJeu]
pre init(largeur,hauteur,maxPas) **require** largeur$\geq 256 \wedge$ hauteur$\geq 240 \wedge$ maxPas≥ 0
Operators :
pasJeu : [MoteurJeu] × COMMANDE × COMMANDE → [MoteurJeu]
pre pasJeu(M,comGuile,comRyu) **require** ¬estFini(M)
Observations :
[invariants]
$0 \leq$ pasJeuCourant(M) $\leq$ maxPasJeu(M)

$$\text{estFini(M)} \overset{\min}{=} (\text{Personnage::estVaincu(GestionCombat::guile(combat(M)))} \vee \text{Personnage::estVaincu(GestionCombat::ryu(combat(M)))} \vee \text{pasJeuCourant(M)=maxPasJeu(M))}$$

$$\text{resultatFinal(M)} \overset{\min}{=} \begin{cases} \text{GUILEGAGNANT si Personnage::estVaincu(GestionCombat::ryu(combat(M)))} \\ \qquad \wedge \neg\text{Personnage::estVaincu(GestionCombat::guile(combat(M)))} \\ \text{GUILEPERDANT si Personnage::estVaincu(GestionCombat::guile(combat(M)))} \\ \qquad \wedge \neg\text{Personnage::estVaincu(GestionCombat::ryu(combat(M)))} \\ \text{NULLE sinon} \end{cases}$$

[init]
maxPasJeu(init(l,h,m))=m
pasJeuCourant(init(l,h,m))=0
combat(init(l,h,m))=GestionCombat::init(l,h)
[pasJeu]
pasJeuCourant(pasJeu(M,cG,cR))=pasJeuCourant(M) +1
combat(pasJeu(M,cG,cR))=GestionCombat::gerer(combat(M),cG,cR)

GestionCombat

service : GestionCombat
use : Personnage
types : boolean, int, String, enum COMMANDE{RIEN, GAUCHE, DROITE, FRAPPE}
observators :
const largeur : [GestionCombat] → int
const hauteur : [GestionCombat] → int
guile : [GestionCombat] → Personnage
ryu : [GestionCombat] → Personnage
estFrappé : [GestionCombat] × String → boolean
pre estFrappé(C,id) **require** id="Guile" ∨ id="Ryu"
estGelé : [GestionCombat] × String → boolean
pre estGelé(C,id) **require** id="Guile" ∨ id="Ryu"
position : [GestionCombat] × String → int
pre position(C,id) **require** id="Guile" ∨ id="Ryu"
collisionGauche : [GestionCombat] × String × String → boolean
pre collisionGauche(C,id1,id2) **require** (id1="Guile" ∧ id2="Ryu") ∨ (id1="Ryu" ∧ id2="Guile")
collision : [GestionCombat] → boolean
Constructors :
init : int × int → [GestionCombat]
Operators :
gerer : [GestionCombat] × COMMANDE × COMMANDE → [GestionCombat]
Observations :
[invariants]
$$\text{collisionGauche(C,id1,id2)} \overset{\min}{=} \Big(\text{-d} \leq \text{position(C,id2) - position(C,id1)} \leq \text{d+1} \Big) \wedge \Big(\text{d} = \text{Personnage::largeur(guile(C))/2 + Personnage::largeur(ryu(C))/2} \Big)$$
$$\text{collision(C)} \overset{\min}{=} \text{collisionGauche(C,"Guile","Ryu")} \vee \text{collisionGauche(C,"Ryu","Guile")}$$
[init]
largeur(init(l,h))=l ∧ hauteur(init(l,h))=h
guile(init(l,h))=Personnage::init("Guile",13,51,16,1664)
ryu(init(l,h))=Personnage::init("Ryu",13,51,16,1664)
estFrappé(init(l,h),id)=false
estGelé(init(l,h),id)=false
position(init(l,h),"Guile")=10 + Personnage::largeur(guile(C))/2 + 1
position(init(l,h),"Ryu")=l - 10 - Personnage::largeur(ryu(C))/2 - 1
[gerer]
$$\text{guile(gerer(C,cG,cR))} = \begin{cases} \text{guile(C) si } \neg\text{estFrappé(gerer(C,cG,cR),"Guile")} \\ \text{Personnage::retrait(guile(C),Personnage::force(ryu(C))) sinon} \end{cases}$$
$$\text{ryu(gerer(C,cG,cR))} = \begin{cases} \text{ryu(C) si } \neg\text{estFrappé(gerer(C,cG,cR),"Ryu")} \\ \text{Personnage::retrait(ryu(C),Personnage::force(guile(C))) sinon} \end{cases}$$
estFrappé(gerer(C,cG,cR),"Guile") = ¬estGelé(C,"Ryu") ∧ cR=FRAPPE ∧ collision(C)
estFrappé(gerer(C,cG,cR),"Ryu") = ¬estGelé(C,"Guile") ∧ cG=FRAPPE ∧ collision(C)
estGelé(gerer(C,cG,cR),"Guile") = estFrappé(gerer(C,cG,cR),"Guile") ∨ (¬estGelé(C,"Guile") ∧ cG=FRAPPE)
estGelé(gerer(C,cG,cR),"Ryu") = estFrappé(gerer(C,cG,cR),"Ryu") ∨ (¬estGelé(C,"Ryu") ∧ cR=FRAPPE)

//Attention : concernant la correction de l'examen 1, on omettera dans ce qui suit les appels à Personnage::largeur car c'est fastidieux...
//... mais cela ne veut pas dire qu'on a le droit de faire pareil dans le projet de l'UE.

$$\text{position(gerer(C,cG,cR),"Guile")} = \begin{cases} \text{min\{position(C,"Guile")+64, largeur(C)\} si estFrappé(gerer(C,cG,cR),"Guile")} \\ \qquad \wedge \text{ collisionGauche(C,"Guile","Ryu")} \\ \text{max\{position(C,"Guile")-64, 0\} si estFrappé(gerer(C,cG,cR),"Guile")} \\ \qquad \wedge \text{ collisionGauche(C,"Ryu","Guile")} \\ \text{max\{position(C,"Guile")-1, 0\} si } \neg\text{estFrappé(gerer(C,cG,cR),"Guile")} \\ \qquad \wedge \neg\text{collisionGauche(C,"Guile","Ryu")} \wedge \text{cG=GAUCHE} \\ \text{min\{position(C,"Guile")+1, largeur(C)\} si } \neg\text{estFrappé(gerer(C,cG,cR),"Guile")} \\ \qquad \wedge \neg\text{collisionGauche(C,"Ryu","Guile")} \wedge \text{cG=DROITE} \\ \text{position(C,"Guile") sinon} \end{cases}$$

$$\text{position(gerer(C,cG,cR),"Ryu")} = \begin{cases} \text{min\{position(C,"Ryu")+64, largeur(C)\} si estFrappé(gerer(C,cG,cR),"Ryu")} \\ \qquad \wedge \text{ collisionGauche(C,"Ryu","Guile")} \\ \text{max\{position(C,"Ryu")-64, 0\} si estFrappé(gerer(C,cG,cR),"Ryu")} \\ \qquad \wedge \text{ collisionGauche(C,"Guile","Ryu")} \\ \text{max\{position(C,"Ryu")-1, 0\} si } \neg\text{estFrappé(gerer(C,cG,cR),"Ryu")} \\ \qquad \wedge \neg\text{collisionGauche(C,"Ruy","Guile")} \wedge \text{cR=GAUCHE} \\ \text{min\{position(C,"Ryu")+1, largeur(C)\} si } \neg\text{estFrappé(gerer(C,cG,cR),"Ryu")} \\ \qquad \wedge \neg\text{collisionGauche(C,"Guile","Ryu")} \wedge \text{cR=DROITE} \\ \text{position(C,"Ryu") sinon} \end{cases}$$