

420-V31-SF - PROGRAMMATION DE JEUX VIDÉO III

TRAVAIL PRATIQUE #3

LE SIÈGE DE LA TOUR DU SORCIER

PONDÉRATION : 20%

OBJECTIFS PÉDAGOGIQUES

Ce travail vise à familiariser l'étudiante ou l'étudiant avec les objectifs suivants :

- 016T - Appliquer une approche de développement par objets
- 0176 - Apporter des améliorations fonctionnelles à une application
- 017C - Concevoir et développer une application dans un environnement graphique.

De même, il permet à l'étudiante ou à l'étudiant d'appliquer les standards de conception, de documentation et de programmation.

MISE EN CONTEXTE ET MANDAT

Il s'agit du travail synthèse du cours de programmation de la troisième session du programme de DEC en informatique – programmation de jeu vidéo. Il vise à faire la synthèse des compétences acquises durant la session.

Les étudiants peuvent suivre le modèle présenté, ou alors présenter leur propre design de projet, à condition que le tout respecte les critères au niveau des objectifs de maîtrise des compétences du cours.

L'étudiant devra donc, suite aux spécifications de l'application, définir une architecture orientée objet à l'aide de langage UML. Ici une simple architecture de base sera requise.

Par la suite, dans un environnement C++ / SFML, l'étudiant devra développer un jeu complet et en assurer la qualité à la fois par des tests d'utilisation et l'utilisation d'assertion ou de tests unitaires.

SOLUTION À PROJETS MULTIPLES.

Ce travail est fait en parallèle avec le cours d'Algorithmique II. Le travail que vous ferez dans le cours d'Algorithmique II sera réutilisé et mis en preuve dans ce travail. Vous devrez donc créer trois projets dans la solution de ce jeu.

- Le projet des algorithmes de pile, file et liste doublement chaîné et circulaire.
- Le projet de test de ces mêmes structures de données
- Le projet de jeu en tant que tel.

SPÉCIFICATIONS DE L'APPLICATION DE BASE

Vous allez créer un jeu stratégie. La version proposée sera tout de même une version simplifiée puisqu'un jeu de stratégie temps réel demande normalement beaucoup de connaissances et de travail au niveau du pathfinding et de l'intelligence artificielle. Néanmoins, ce jeu vous permettra de développer vos compétences au niveau de l'héritage

multiple et des patrons de conception observateur et composite. De plus elle mettra en application concrète les structures de données que vous aurez développé en algorithmique.

Vous n'êtes pas obligé de suivre la version de base telle que proposée, mais dans ce cas, vous devrez remplir un mini-document de conception pour illustrer votre jeu et surtout dans quelles circonstances seront utilisées les différentes contraintes ci-dessous.

Dans tous les cas, votre jeu pourra en être un de type stratégie temps-réel, aréna de bataille, "tower defense" ou même moba. Sachez seulement que pour garder les choses simples le jeu devrait être asymétrique au niveau des forces (vous aurez des sorts / effets spéciaux et un peu plus de choix de décision; et pas l'autre camp. Donc le camp adverse devra être un peu plus nombreux. Vous pouvez faire un jeu qui se joue exclusivement à deux joueurs. Mais il faudra vous débrouiller sans la souris dans votre design, à moins que le jeu soit vraiment asymétrique au niveau des forces, et même des contrôleurs.

RÉFÉRENCES

Quelques exemples pertinents pour chaque style de jeu, évidemment, ces listes sont loin d'être exhaustives.

Stratégie temps réel

Starcraft

Dawn of War

Command and Conquer

Multiple Online Battle Arena

Defense of the Ancients

League of Legends

Heroes of the Storm

Aréna de Bataille

Clash Royale

Miragine War

Tower Defense

Kingdom Rush

Toy Defense 2

TP3 – LE SIÈGE DE LA TOUR DU SORCIER

Vous étiez le sorcier royal, mais le roi n'a pas apprécié que vous fassiez péter son château lors de votre dernière expérience qui a mal tournée. Il a donc juré de mettre votre tête sur une pique. Vous vous êtes donc réfugié dans votre tour. Le roi a bâti trois tours près de votre domaine et envoie ses troupes à l'assaut de la vôtre. Vous avez bien moins de troupes que lui, mais vous êtes plus brillant que lui et vous avez votre magie pour vous aider.

Vous gagnez la partie lorsque vous éliminez les trois tours adverses. Évidemment, vous perdez si votre tour est détruite.

Le jeu de base

À la base, il s'agit d'un jeu d'Aréna de Bataille. Donc, les soldats se déplaceront vers le camp adverse et attaqueront une tour ennemie s'ils s'y rendent. Si un ennemi arrive à proximité relative (100 pixels environ) d'un soldat, le soldat va se diriger vers son ennemi et l'attaquer. S'il n'y a plus d'ennemi à proximité, le soldat reprend son chemin.

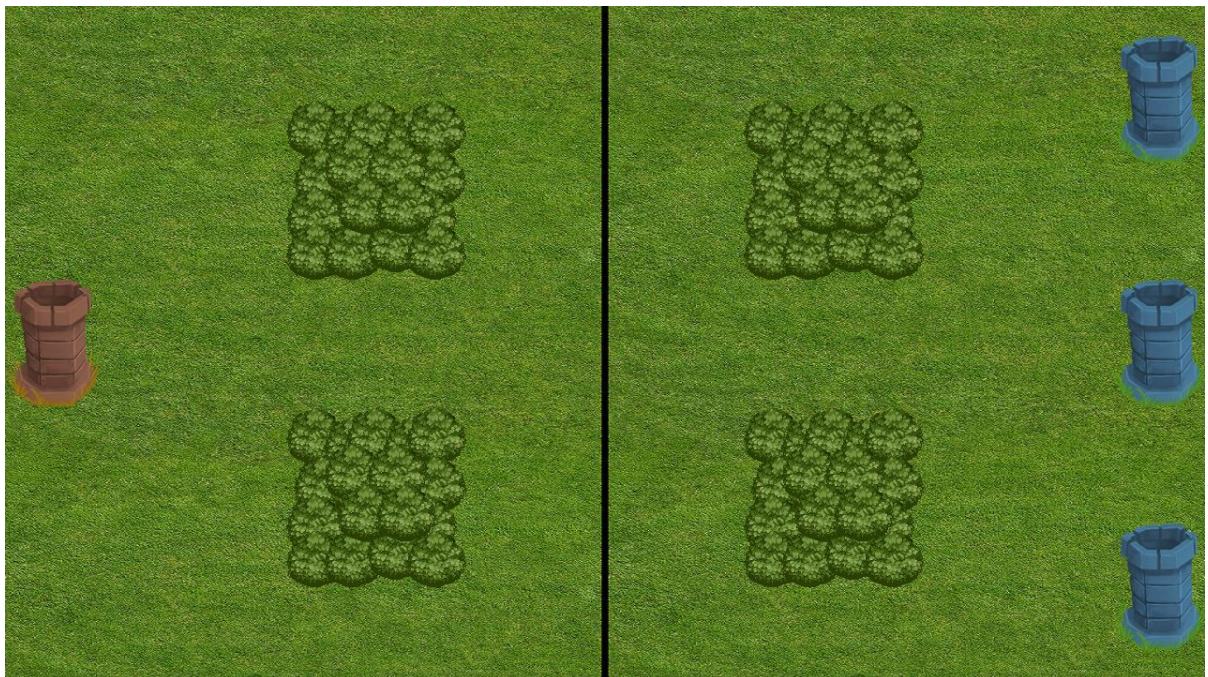
Un soldat se dirige toujours vers l'ennemi (ou la tour) le plus proche, mais une fois qu'il attaque une cible, il garde la même cible jusqu'à ce qu'il l'ait éliminé.

Les soldats, une fois sortis de leur tour, vont choisir un point au hasard sur la ligne médiane (pour que les soldats ne s'empilent pas), et une fois cette ligne atteinte, ils vont se diriger vers la tour adverse la plus proche. Si leur tour-cible est détruite, ils s'en choisiront une autre. Le camp qui détruit toutes les tours adverses remporte la partie.

(Pour les points sur la ligne du milieu : Vous pouvez mettre 5 à 7 points fixes sur la ligne comme point de destination, ou mieux encore, donner un choix total sur la ligne. Évitez par contre les sections trop près des extrémités haut et bas de l'écran.)

Notez qu'à la base, le jeu ne comprend pas de terrain difficile ou infranchissable.

Le joueur humain peut donner une destination plus précise à un soldat ou à un groupe de soldats, à la manière d'un jeu de stratégie temps réel. Une fois le point atteint, le soldat reprend son chemin habituel. S'il est passé du côté adverse du champ de bataille, sa prochaine destination est une tour.



Génération des soldats

Le joueur humain (sorcier) dispose de deux (2) points de ressource par seconde et peut sélectionner le type de soldat dans un sous-écran de sélection. Le joueur humain demande le "spawning" du dit soldat à l'aide d'une commande et s'il dispose d'assez de points de ressources, le soldat est généré.

Le joueur AI dispose de quatre (4) points de ressource par seconde. Chaque tour a une file de quatre soldats générée aléatoirement. Ces files sont au départ cachées. Chaque tour génère un soldat à tour de rôle. Dès que le camp du roi a assez de point de ressources, le soldat est généré. Puisque le nombre de points de ressources pour le Roi est toujours le même, donc s'il reste deux tours au roi, chaque tour générera plus rapidement ses soldats et une fois qu'il ne restera qu'une tour, celle-ci générera ses soldats très rapidement.

Par contre, un combat plus concentré rendra le combat plus avantageux pour le sorcier puisque ses sorts à zone d'effets devraient être plus efficaces.

Sorts

Le joueur peut choisir ses sorts dans un sous-écran de sélection. Au départ, le joueur n'a qu'un sort (Éclair). À mesure que des ennemis sont vaincus, ceux-ci peuvent laisser tomber (15 à 20% de chance environ) des sources magiques qui génèrent des sorts. Le joueur peut les récupérer en cliquant dessus et les ajouter à sa liste de sort.

Chaque source permet d'utiliser un sort précis à trois reprises. Si le joueur trouve une autre fois la même source, ses utilisations augmentent. Un sort qui n'a plus d'utilisation est retiré de la liste.

Les sorts ne sont pas lancés gratuitement. Le joueur dispose d'un point de magie par seconde. **Donc un sort consomme une utilisation ET des points de magie.**

Chaque sort est lancé à une position en cliquant à l'écran. Si une cible potentielle est assez proche, elle peut être affectée par le sort.

Notez que par un caprice de notre univers de jeu, seul le dernier sort lancé prend effet. L'avant-dernier est "mis en pause" et le dernier prend effet. Quand ce même dernier est terminé l'avant-dernier reprend et ainsi de suite (principe de pile), les sorts instantanés doivent tout de même empilés et dépilés immédiatement.

Affichage, animation et collision des soldats

Quelques sprites sont fourni avec cet énoncé; bien entendu il en manque pour tout illustrer ce projet. Vous êtes libre d'utiliser tous les assets de votre choix.

Le fond d'écran est en 1280 x 720, par contre il ne laisse pas de place à l'interface si vous choisissez cette résolution. Vous pouvez utiliser une résolution plus grande, ou tailler ce fond pour faire place à l'interface.

Gardez l'animation des soldats et des sorts minimale, mais illustrez-les tout de même. Une manière simple serait de faire tirer un projectile à chaque soldat qui en attaque un autre. Même un personnage qui a une portée de "corps à corps" lancerait un projectile à très courte portée.

Autre méthode, faire afficher une arme au-dessus du personnage qui attaque et le montant des dommages sur celui qui est attaqué.

Les soldats devraient pouvoir collisionner entre eux et avec les tours.

Pour les sorts, vous devriez trouver des logos pour les "sources". Ces logos prendront place dans la liste des sorts et quand un soldat sera affecté par un sort nous verront un petit logo de ce sort affiché près de lui.

Interface et commandes

La sélection des soldats et des sorts utilisera une liste doublement chaînée et circulaire développé en algorithme. Vous devriez avoir tous vos effets illustrés dans des listes qui s'agrandiront à mesure qu'on y ajoute des éléments. L'élément sélectionné est entouré par un cadre. Si vous arrivez à une extrémité, la sélection va de l'autre côté. Vous devriez manipuler vos listes à l'aide du clavier. (ex : A et S pour les soldats, Z et X pour les sorts)

Commandes suggérées (ce sont vraiment des suggestions)

Déclencher la sortie d'un soldat devrait se faire avec une touche du clavier (ex : spacebar) ou en cliquant sur la tour.

Manipuler des troupes devrait se faire à la manière d'un jeu de stratégie standard. Cliquer sur un soldat ou sélectionner un groupe avec le bouton gauche de la souris. Désigner la cible avec le bouton droit.

Cliquer sur une source magique permet de la récupérer.

Le lancer de sort devrait se faire avec l'appui d'un bouton du clavier (ex : Ctrl gauche) plus un clic de la souris.

Le dernier soldat ou groupe de soldat sélectionné pourrait lancer son habileté spécial avec un bouton à l'écran ou une touche du clavier (Alt gauche)

STRUCTURES DE DONNÉES

Le projet se doit d'utiliser piles files et listes produites dans votre TP. Chaque structure doit être utilisée au moins une fois, et vous devrez avoir cinq utilisations en tout.

Dans " Le siège de la tour du sorcier" leur utilisation est la suivante :

File : chaque tour a une file de quatre soldats qui peuvent être générés. Quand un soldat sort de la tour, un nouveau est généré en queue de file. Un sort permet de révéler peu à peu les éléments de la file de chaque tour.

Liste 1: Les soldats que l'on peut utiliser sont placés dans une liste doublement chaînée et circulaire.

Liste 2 : Les sorts que l'on peut utiliser sont placés dans une liste doublement chaînée et circulaire. Un sort trouvé qui était non-présent est placé dans la liste alors qu'un sort qui n'a plus d'utilisation est retiré.

Pile 1 : Les chemins des soldats. Quand un soldat est créé, les destinations sont déjà empilées. Une tour est d'abord empilée, puis les autres s'il en reste. Puis est empilé un point à mi-chemin du champ de bataille. Si le joueur donne à un soldat une "destination custom" elle est empilée sur le dessus. Quand la position custom est dépilée, si le joueur est rendu dans le camp adverse et que sa position de mi-chemin était toujours présente, elle est immédiatement dépilée. Un soldat sans cible est immobile.

Pile 2 : Effet des sorts. Un effet de sort sur un soldat est empilé. S'il en subit un autre, celui-ci est aussi empilé et agit sur le soldat. Seul le sort sur le dessus de la pile peut agir. Une fois le délai du sort terminé, il est dépilé et le sort précédent reprend acte.

PATRONS DE CONCEPTION

Le projet se doit d'utiliser des patrons de conception telle que vu dans le cadre du cours. Chaque structure doit être utilisée au moins une fois (deux fois pour observateur), et vous devrez avoir cinq utilisations en tout.

Fabrique statique : Tous les soldats sont générés à l'aide de fabriques statiques.

Observateur 1 : (Sujet: sort / Observateurs : soldats et tours) Quand un sort est lancé. Il est lancé à une position et peut affecter à la fois les soldats et les tours. Chaque observateur doit vérifier s'il est à distance pour être affecté par un sort. S'il l'est, c'est à lui de traiter sa réaction au sort. La collection des observateurs devrait être gérée par la game elle-même, mais quand un sort est créé, il reçoit un pointeur sur cette liste.

Attention : Si vous avez un design custom, notez que pour le pattern observateur vous devriez avoir une sélection diversifiée d'observateurs qui répondent. Ici, certains effets de sort affectent tous les soldats, d'autres juste un camp, et d'autres certaines sous-classes précises.

Observateur 2 : (Sujet: tours / Observateurs : soldats). Quand une tour est détruite, elle lance un événement. Chaque soldat qui a cette tour comme cible actuelle doit immédiatement la dépiler et passer à la cible suivante. Ceux qui l'auront toujours dans la pile et que suite à un dépilement l'ont comme tour cible plus tard devront vérifier si elle est toujours active (ce qui ne sera plus le cas) et si ce n'est pas le cas, la dépiler immédiatement.

Attention : Pour simplifier votre projet et éviter des bugs, il serait plus simple de ne pas effacer un tour de la mémoire quand celle-ci est détruite, mais plutôt lui mettre un état détruit.

Composite 1: Déplacement d'un groupe de soldat à une cible déterminée par le joueur comme dans l'ensemble "Mist of Souls" ou dans n'importe quel jeu de stratégie en temps réel. Un groupe de soldat est un composite. Dans notre collection de composants on peut seulement ajouter des soldats ou la vider.

Composite 2: Suite à un sort de clonage, un composite de deux soldats identiques à lui-même est créé. Le soldat original est placé dans le composite alors que les autres sont ajoutés. Mettre les soldats clonés à proximité du soldat original au hasard. Répéter jusqu'à ce qu'ils soient dans une position sans collision.


LISTE DES TOURS ET SOLDATS

Notez que chaque soldat dispose d'une habileté spéciale, qui justifie sa propre sous-classe; bref qui justifie qu'il ne soit pas un simple soldat avec des statistiques différentes. Chaque habileté spéciale prend environ 10 secondes à recharger. Le joueur pourrait sélectionner un soldat ou un groupe de soldat et déclencher l'habileté spéciale avec la même commande, peu importe le type de soldat

Ici vous avez un exemple de sept (7) types de soldat. Vous n'êtes pas obligé de créer tous les types présents; 3 est un minimum, et 5 est très acceptable.

Vous pouvez ajuster la balance selon les tests que vous ferez; les valeurs présentées sont tout à fait modifiables.

Soldats

TypeSoldat	Sprite	Vitesse	Points de vie	Portée	Dommages	Ressource	Habileté spéciale
Tour		0	2400	0	0	0	aucune
Archer		3	125	50	35	3	Retraite stratégique
Lancier		4	185	10	55	2	Javelot
Guerrier léger		3	360	5	50	3	Milles lames
Guerrier lourd		3	720	5	70	5	Mur d'acier
Barbare		3	600	5	100	6	Berserk
Cavalier		5	750	10	150	10	Charge
Magicien		3	280	50	200	10	Soin

Notez que les soldats attaquent environ une fois aux deux secondes.

La vitesse est en sprite par rafraichissement

La portée est en pixel

Les dommages peuvent être fixe, mais pour un effet plus random, chaque coup portée pourraient être entre 51 et 150% de la valeur de base.

Ressource est le nombre de points de ressources que demande le soldat.

Retraite stratégique: L'Archer recule immédiatement vers ses propres lignes pendant 3 secondes à la vitesse de 5. Si les archers passent la frontière à reculons, ils gardent quand même leur tour actuelle comme cible. Ils ne peuvent pas reculer derrière l'écran.

Javelot: La portée du lancier augmente à 40 durant 3 secondes

Milles lames: Le guerrier léger double ses dommages durant 3 secondes.

Mur d'acier: Le guerrier lourd prend demi-dommage durant 3 secondes.

Berserk: Quand le barbare frappe, toutes les unités ennemies à portée prennent un coup durant 3 secondes

Charge: La vitesse augmente à 8 et les dommages sont doublés durant 3 secondes

Soin: L'effet d'un sort de soin est lancé avec comme point central le magicien lui-même.

AI de base: Vous pourriez faire en sorte que quand un ennemi rencontre un soldat du joueur, celui-ci peut déclencher son habileté après 1 à 3 secondes après la rencontre.

Sorts

Nom du sort	Coût	Camp	Classe	Temps	Effet
Éclair	2	Tous	Tous	Instantané	80 points de dommage à tous les soldats à 25 pixels de rayon du point d'impact
Boule de feu	4	Tous	Tous	Instantané	250 points de dommage à tous les soldats à 25 pixels de rayon du point d'impact
Météore	6	Tous	Tous	Instantané	500 points de dommage à tous les soldats à 35 pixels de rayon du point d'impact
Gel	4	Tous	Tous	3 secondes	Tous les soldats à 30 pixels du point d'impact sont complètement paralysés pour la durée du sort.
Fatigue	2	Ennemi	Tous	4 secondes	Les soldats ennemis à 40 pixels du point d'impact perdent 1 de vitesse de déplacement et leur dommage réduit de 40%
Vigueur	2	Ami	Tous	4 secondes	Les soldats amis 40 pixels du point d'impact gagnent 1 de vitesse de déplacement et leur dommage augmenté de 40%
Clonage	4	Ami	Un soldat	5 secondes	Création de deux clones du soldat visé (voir description composite). Demande de la précision. Si le soldat est raté, le sort est perdu.
Poison	3	Ennemi	Tous	10 secondes	Tout soldat à 40 pixels du point d'impact est empoisonné. Un soldat empoisonné perd 15 points de vie à la seconde.
Guide	2	Ami	Archers et lanciers	7 secondes	Les archers et lanciers ami à 30 pixels du point d'impact gagnent 30 pixels de portée
Soin	3	Ami	Tous	Instantané	Tous les soldats à 30 pixels du point d'impact sont soignés complètement
Affutage	4	Ami	Guerriers et barbares	7 secondes	Les dommages de base des guerriers et barbares est doublé
Divination	1	Ennemi	Tour ennemi	Instantané	Révèle une case d'une file d'ennemis d'une tour donnée.

Notez que le joueur commence la partie avec 9 éclairs

Éclair, boule de feu et météore font demi-dommage contre les tours.

CONTRAINTES ET CONSEILS DE DÉVELOPPEMENT

- Dans les méthodes, favoriser les références ou les pointeurs pour les objets. Ne pas faire de copie d'objets lourds.
- Classes: faites les déclarations dans le .h, et les définitions dans le cpp.
- Attention aux erreurs de linkage: ne pas laisser une classe déclarée non définie, définissez la méthode et faites le "return" minimum pour que la méthode compile.
- Favorisez les constantes plutôt que les valeurs inscrites, et déclarez-les statiques si elles portent une valeur universelle pour chaque instance de la classe.
- Documentez votre code. Les attributs et méthodes doivent être commentés (triple slash). Les algorithmes complexe et ou critiques peuvent être commentés dans le code.
- Gérez correctement votre mémoire. Bons destructeurs et mettez vos destructeurs virtuels dès qu'il y a situation d'héritage.
- Utilisation systématique de const, à la fin des méthodes qui ne modifie pas l'objet courant (dont les get) et const sur les paramètres en entrée que l'on ne veut pas modifier (donc sur la grande majorité des set)
- Pas de contrainte pour les assets, vous pouvez utiliser ceux de votre choix. Pensez à travailler avec des placeholders au besoin. Vous placerez les bons assets plus tard.
- Vous pouvez travailler en héritant de sprite ou en séparant complètement la logique de jeu de l'affichage, à votre choix.
- Ne vous attardez pas au début sur la qualité de production du projet. Faites une bonne base fonctionnelle avant tout. La qualité de production du projet est facile à mettre en place une fois la base du projet fonctionnelle.

DOCUMENTS À FOURNIR

Au début du projet

Pour vous inspirer, ce projet peut être relativement libre, tant que le type de jeu reste dans les barèmes présenté. Si c'est bien le cas, vous devez faire un document de conception en au moins une page (pensez au moins à deux) pour décrire le plus exhaustivement votre projet. Ceux qui reprennent le projet tel que présenté sont épargnés de cette tâche, et ceux qui s'y basent mais font des différences doivent le mentionner également.

Un diagramme de classe de base doit aussi être fourni avec seulement les classes et les relations (ne pas y mettre d'attributs et de méthodes) mais ça vous fera une bonne réflexion sur la direction à prendre.

À la fin du projet

Vous devez fournir un journal de projet qui va démontrer le déroulement du développement de votre travail pratique. Voici que qu'il devrait comprendre.

- Page titre et table des matières.
- Votre document de conception et votre diagramme de classe préliminaire.
- Description finale du jeu (pourrait être différent de ce que vous avez prévu au départ)
- Liste des commandes.
- **Description des assets** : quel asset représente quoi dans le projet.
- **Un log de tâche** : à chaque fois qu'un des membres de l'équipe travail sur le projet durant une journée donné, il doit inscrire la date, le temps de travail total et les tâches réalisées.
- Ce journal est absolument obligatoire. Ne pas le remettre entraine une pénalité.
- Faites-le à mesure que le projet avance.
- **Je me répète, faites-le à mesure.**
- **Juste pour être absolument certain : FAITES-LE À MESURE**

BONUS

Pour qu'un élément bonus puisse s'appliquer un total de 80% de la note de base est nécessaire avant que les bonus soient pris en compte, et tous les éléments des objectifs suggérés doivent être fait Faites une bonne base avant tout.

Voici quelques suggestions de bonus potentiels :

- Qualité de production supérieure
- AI avancée
- Obstacles et pathfinding
- Gestion du terrain (couvert, ralentissement, etc.)
- Éléments de menus : Écrans de début et de fins interactives, menus d'options qui auront des effets dans le jeu
- Mapping des commandes que l'utilisateur peut changer
- Système sonore élaboré.
- Tests unitaires (compte même sans le 80%)
- Etc.

CONTEXTE DE RÉALISATION ET DÉMARCHE DE DÉVELOPPEMENT

Ce travail pratique doit être réalisé **en équipe de deux étudiants**. Les équipiers seront les mêmes dans le cours d'algorithmique.

Pour faciliter la coordination entre les deux professeurs sur le projet et pour assurer un bon suivi, le projet devra être obligatoirement être placé et entretenu sur un **dépôt SVN** fourni à chacun des membres de l'équipe. Le manque d'utilisation du dépôt pourrait être pénalisé sur le premier critère de notation du TP.

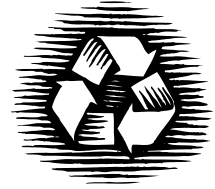
À partir de la semaine 14, votre projet devrait toujours être compilable à la fin de chaque cours. Si ce n'est pas le cas, en avertir le professeur et rectifier la situation le plus rapidement possible.

Biens livrables :

- Projet avec code source de l'application (code bien documenté, y compris le document de projet qui doit être aussi sur le dépôt)
- Application fonctionnelle
- Récupération du dépôt SVN le **dimanche le 18 décembre à minuit.**
- Vous pouvez vous "backer" avec une remise sur LEA, mais la première récupération sera votre projet sur SVN.

MODALITÉS D'ÉVALUATION

- Tous les **biens livrables** devront être **remis à temps** et selon les modalités spécifiées.
- Dans l'éventualité où vous récupérez du code existant ailleurs (internet, MSDN, etc), vous devez clairement indiquer la source ainsi que la section de code en question. Tout travail plagié ou tout code récupéré d'une source externe et non mentionnée peut entraîner la note zéro (0) pour l'ensemble du travail.



Grille d'évaluation du travail pratique #3

Éléments génériques	
Critères d'évaluation	
<ul style="list-style-type: none"> - Utilisation correcte du C++. Code clair et bien documenté. Exploitation judicieuse des possibilités du langage. Algorithmes compréhensibles. Division du code en .h et .cpp fait correctement. <p>Modèle orienté objet correcte. Utilisation optimale des classes. Respect de l'encapsulation. Modèle cohérent. Bonne utilisation des const et des constantes.</p>	15%
<ul style="list-style-type: none"> - Journal de projet (pénalité de 5% supplémentaire si non remis). 	5%
Programmation de jeux vidéo III	
<ul style="list-style-type: none"> - Jeu fonctionnel suivant les spécifications demandées ou selon entente suivant votre concept. 	25%
<ul style="list-style-type: none"> - Utilisation judicieuse et correcte de l'héritage 	10%
<ul style="list-style-type: none"> - Utilisation judicieuse et correcte du patron observateur 	25%
<ul style="list-style-type: none"> - Utilisation judicieuse et correcte des patrons fabrique et composite 	10%
<ul style="list-style-type: none"> - Qualité de production : on doit rendre le jeu professionnel: <ul style="list-style-type: none"> *Collisions correctes *Affichage graphique sans bug *Quelques animations simples (exemple : des explosions) * Utilisation de quelques éléments sonores * Interface intéressant de manipulation des soldats et sorts * Écran de départ et de fin. * Tout élément pertinent qui ajoute de la qualité, de la "production value" 	10%
TOTAL	100%
Bonus discrétionnaires: ajout potentiel de 10% pour une note maximale de 100%	