Groupe SIC du Cycle d'orientation

PRESENTATION D'UNE SEQUENCE D'APPRENTISSAGE

Titre de la séquence : JEU DE SQUASH.

But de la séquence : réalisation d'un jeu de squash.

Auteur(s) (nom et collège) : Robert CHALMAS / GO.

Logiciel(s) utilisé(s) : Delphi.

Durée de la séquence : 6-8 heures

Groupes d'élèves concernés : ❒ 7ème ❒ A ❒ B ❒ C

☒ 8ème ☒ A ❒ B ☒ H

Domaines du plan d'études concernés :

- ❒ **Connaître les systèmes**
- ☒ **Programmer l'ordinateur**
- ❒ **S'exprimer à l'aide des outils multimédias**
- ❒ **Correspondre et se documenter**
- ❒ **Manipuler et présenter des données**

Contexte général de la séquence

A. *Environnement pédagogique*

(Cocher une ou plusieurs cases)

❖ **Contenu de la séquence**

❒ disciplinaire ☒ pluridisciplinaire ❒ transdisciplinaire

❖ **Type d'activité**

❒ traitement de texte ❒ tableur ❒ présentation d'un document

❒ édition de pages html, construction de site

❒ dessin, retouches d'images, animation

❒ élaboration d'un document multimédia

❒ recherche et activités didactiques sur Internet ❒ messagerie électronique

☒ programmation

❖ **Mode de travail des élèves**

☒ individuel ❒ par 2 ☒ en réseau

❒ utilisation des aides des logiciels ❒ utilisation d'un cahier

❖ **Type de pédagogie**

❒ différenciée ☒ par projet* ❒ par résolution de problèmes

❒ socio-constructivisme* ❒ behaviorisme* ❒ pédagogie de maîtrise*

❒ ……

❖ **Mode d'intervention du maître dans l'apprentissage de l'élève**

❒ apports de connaissances sous forme écrites

☒ démonstrations/explications sur écran

❒ consignes du/des travaux pratiques :

❒ sur papier ❒ sous forme de fichiers électroniques

☒ intervention ponctuelle pour dépanner un élève

B. Savoir-faire et compétences développés par les élèves (selon annexe I du PE)

Compétences minimales :

- savoir implanter des objets d'interface (1 Panel, 2 boutons, 2 Shapes, 1 Timer, éventuellement 1 SpinEdit pour l'affichage du score),
- savoir mettre en place des processus événementiels (programmation de deux boutons: lancer la balle et quitter),
- utiliser une structure de test simple (rebondissement de la balle seulement si elle est arrivée contre un mur),
- savoir gérer des processus parallèles (utilisation d'un composant Timer pour faire avancer la balle et tenir à jour le score),
- savoir interpréter une ligne d'instructions et prédire ce que l'ordinateur va faire (beaucoup utilisé lorsque le rebondissement de la balle ne se passe pas comme attendu !),
- savoir définir une variable, consulter, utiliser et modifier son contenu (notamment: utilisation de la valeur des propriétés Left et Top des deux composants représentant la balle et la raquette pour savoir où ils se trouvent sur la surface de jeu et gérer les rebondissements).

Développements réalisés par certains élèves rapides :

- Réalisation d'un dessin avec l'éditeur d'images de Delphi.
- Affichage de cette image dans un composant Image.
- Réalisation d'une icône avec l'éditeur d'images de Delphi.
- Utilisation de cette icône en tant qu'icône du programme à la place de celle de Delphi.
- Ajout d'un compteur pour afficher le nombre de renvois de la balle réussis.

Partie pratique

1. Objectifs de la séquence

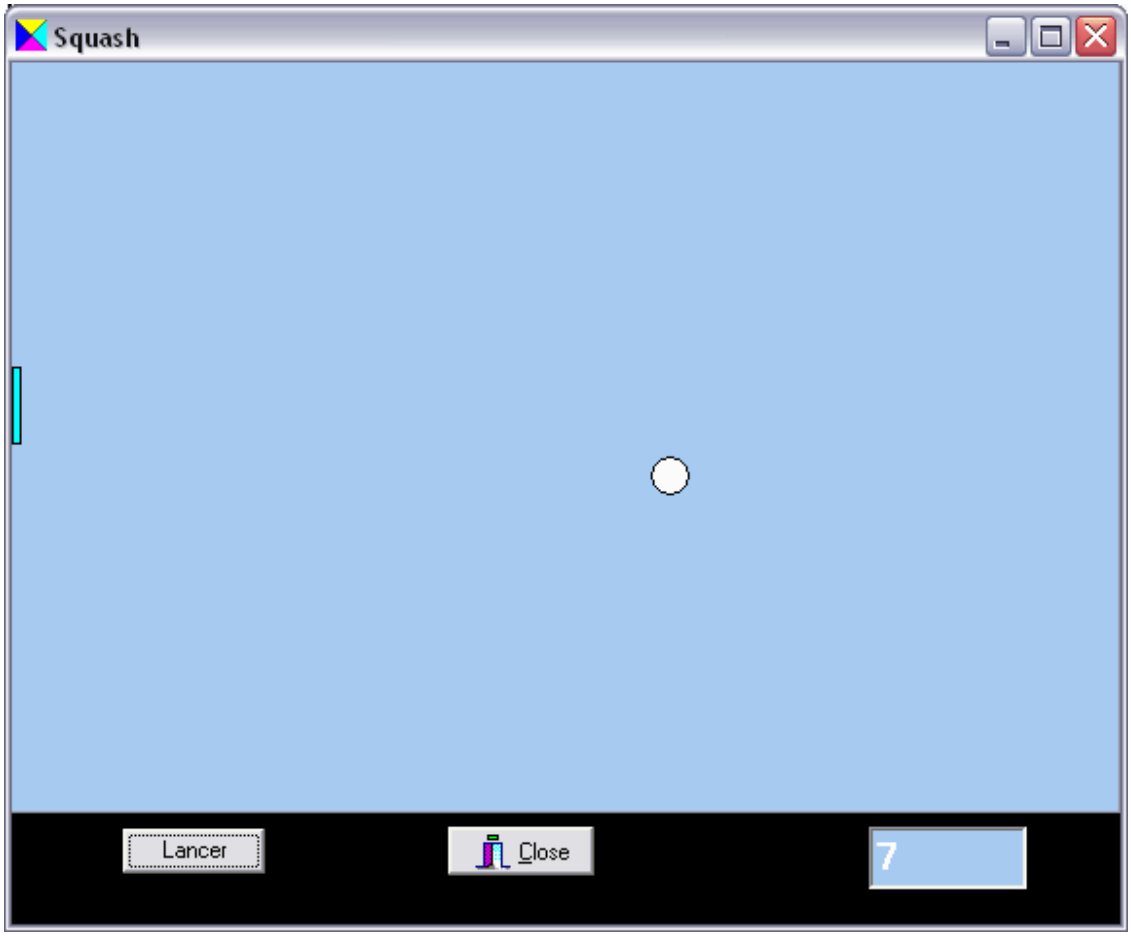
Réaliser une interface graphique, associer des actions à des boutons, réfléchir sur la notion de mouvement (et notamment sa décomposition en deux axes X et Y, comme cela se fait en physique), utiliser cette réflexion pour proposer des solutions pour gérer le rebondissement de la balle, savoir localiser un objet sur la surface de jeu et utiliser cette information pour une prise de décisions (gestion du rebondissement).

2. Déroulement, chronologie de la séquence

1) Démonstration du jeu terminé afin de motiver les élèves et de leur permettre de mieux se représenter ce qui est à faire (composants nécessaires, actions à programmer),
2) Placement des composants sur l'interface graphique (cf. supra),
3) Renommer les 2 composants Shape en Balle et Raquette; adapter leur couleur (propriété Brush.Color) et leur forme (propriété Shape). Placer ces composants à un point initial approprié.

4) Programmer le Timer pour faire avancer la balle horizontalement (par exemple: Balle.Left:=Balle.Left+15) et rechercher un intervalle de répétition donnant un mouvement fluide (propriété Interval; environ 20 ms devrait être un bon choix).
5) Pour que la balle ne se mette pas en mouvement dès le lancement du programme, mettre la propriété Enabled du Timer sur False et programmer le bouton "Lancer la balle" pour basculer cette valeur sur True.
6) Pour que le mouvement ne soit pas seulement horizontal, ajouter aussi un mouvement selon l'axe vertical (par exemple Balle.top:=Balle.top+12),
7) Remplacer les vitesses sous forme numérique par des variables, nommées par exemple vh (vitesse horizontale) et vv.
8) Faire réfléchir un moment les élèves, si désiré par groupes, au problème du rebondissement: que se passe-t-il au niveau de la vitesse quand la balle rebondit sur un mur horizontal ? et sur un mur vertical ? On devrait arriver au résultat que la vitesse reste la même mais la direction s'inverse. Il faudra probablement aider les élèves à réaliser que ceci revient simplement à multiplier la vitesse (horizontale ou verticale selon l'orientation du mur) par -1. Ceci est le point central de ce jeu de squash.
9) Dans la procédure gérant le mouvement, ajouter un test pour voir si la balle a atteint le mur de droite et, si oui, faire rebondir la balle (en multipliant sa vitesse horizontale par -1),
10) Faire de même pour gérer le rebondissement sur les murs du haut et du bas (cette fois c'est la vitesse verticale qui est concernée !),
11) Il reste à gérer le rebondissement sur la raquette. Faire réfléchir les élèves un moment à ce problème. Ils devraient trouver des éléments pertinents, mais incomplets (du genre: les coordonnées X et Y de la balle et de la raquette doivent être identiques); tester le résultat que cela donne (en général: pas de rebondissement),
12) Si les élèves ne trouvent pas l'origine du problème, il faudra probablement les mettre sur la voie: les coordonnées X et Y des deux objets ont peu de chance d'être exactement les mêmes vu que la balle avance par pas de typiquement 10 ou 20 points; de plus, la balle et la raquette ne sont pas des objets ponctuels. Ce point étant difficile pour nos élèves, on peut donner la syntaxe d'un test tenant compte de ces faits, par exemple:
```
if (abs (balle.Left-raquette.Left) < 10) and
   (abs((balle.top+10)-(Raquette.Top+20)) < 20) then {rebondissement}.
```
13) Gérer l'arrêt de la balle si le joueur l'a ratée, en arrêtant le Timer si la coordonnée X est <0.
14) Si le temps le permet, il n'est pas très difficile d'ajouter un compteur de score: définir une variable score, placer un composant pour l'afficher (un SpinEdit est pratique vu que sa propriété Value accepte directement des valeurs numériques) et incrémenter sa valeur de 1 à chaque rebondissement réussi.
15) Pour les élèves rapides, il est toujours bienvenu d'améliorer la présentation du projet en réalisant un dessin qui pourra être placé dans un composant Image et/ou d'une icône qui pourra remplacer l'icône standard de Delphi.

3. **Documents en lien avec cette séquence :**
    - Consignes → lien hypertexte avec le document concerné: Néant.
    - Travaux d'élèves → lien hypertexte avec les travaux des élèves: Néant.
    - Impression d'écran d'un travail d'élève terminé :



Evaluation des élèves

- Mode(s) d'évaluation utilisé(s) : note mise sur le projet en fonction :
    - présentation de l'interface (lisibilité, disposition "raisonnable" des composants),
    - code sans erreurs (donc compilation possible sans corrections),
    - fonctionnement du programme conforme à ce qui avait été demandé, notamment au niveau du rebondissement,
    - participation de l'élève lors des phases de réflexion,
    - autonomie de l'élève,
    - éléments personnels ajoutés par l'élève (dessin, icône,...)

Documents d'évaluation → lien: Néant.

Grille d'évaluation envisagée ou non → lien: Néant.

Bilan de la séquence

- Les objectifs de la séquence ont-ils été atteints ?

Oui. Tous les élèves ont pu réaliser un programme fonctionnel, même si cela a nécessité un temps assez important (7 heures). Le groupe avec lequel ce programme a été testé avait 1 h de SIC à l'année, de ce fait les opérations de mise en train en début d'heure et de sauvegarde en fin d'heure prennent proportionnellement plus de temps qu'avec des blocs de 2h. La plupart des élèves ont cependant pu ajouter le compteur de score, présenté au départ comme facultatif, et dans les deux tiers des cas des composants graphiques facultatifs (image et/ou icône).

- Commentaires sur les difficultés rencontrées.

Il s'agit d'un projet relativement complexe, qui est donc à effectuer plutôt en fin de cours SIC. Il n'est donc pas étonnant que les élèves ne puissent pas tout découvrir tout seuls, notamment au niveau gestion du mouvement de la balle. Le lien avec la physique à ce niveau est toutefois un élément intéressant.

Eléments positifs – Eléments négatifs

Positif.

Les élèves, étant souvent amateurs de jeux vidéo, entrent facilement dans un tel projet, même s'ils en sous-estiment la difficulté. En "mettant la main à la pâte", ils ont l'occasion de se faire une première idée du gros travail nécessaire pour réaliser ce type de logiciels et de comprendre pourquoi les programmes ne sont pas toujours gratuits !

Négatif.

Le projet est assez long à réaliser, de sorte qu'il existe un risque de lassitude chez certains élèves. Si nécessaire, on peut intercaler quelques éléments plus ludiques, par exemple la création d'éléments graphiques (dessins, icône de programme) entre les points les plus difficiles du projet.

- Quelle(s) modification(s) apporteriez-vous à cette séquence ?

Aucune.