Votre deuxième jeu : Run & Jump

Bonjour à tous ! Aujourd'hui, nous nous retrouvons pour concrétiser les notions que vous avez vues lors des derniers cours. Notre jeu sera « Run & Jump », un jeu de plateforme où une boule devra sauter pour ne pas tomber dans les trous.

Réflexions sur le jeu :

Le jeu sera divisé en quelques parties : la gestion du terrain (la map), le saut de la boule, les collisions, et l'affichage. Nous verrons en détail comment programmer ces différents modules. En effet, avant de commencer la partie pratique, c'est à dire sur la calculatrice, il est nécessaire de mettre sur papier les grandes lignes de l'algorithme afin de ne pas se laisser surprendre par d'éventuels bugs ou de rendre illisible le code car on a oublié une partie.

L'initialisation et la boucle principale :

Le jeu sera constitué d'une boucle qui tournera tant que l'on n'est pas tombé dans un trou. On initialisera à zéro la distance parcourue, la valeur du Getkey et la position sur l'axe des Y du joueur.

```
0→D // la distance qu'a parcouru le joueur
4→P // la position en Y du joueur
0→G // la valeur du Getkey
While 1
D+1→D // on augmente la distance
… // Les instructions principales
WhileEnd
```

La gestion du terrain :

Le terrain sera constitué d'une succession de 21 « T », qui symboliseront le sol. Ensuite, des espaces ajoutés aléatoirement dans le sol symboliseront les trous au-dessus desquels il faudra sauter. A chaque « frame », c'est à dire à chaque tour de boucle principale, nous supprimeront le bout de sol qui sortira de l'écran, et nous ajouteront soit un « T », soit un espace.

Nous allons utiliser, pour l'aléatoire, la fonction « RandInt » qui génère un nombre entier aléatoire entre les deux bornes qu'on lui donne. Elle se trouve dans le menu : [OPTN] / [F6] / [F3] (PROB) / [F4] (RAND) / [F2] (Int)

Le problème, c'est qu'il ne faut pas qu'il y ait deux trous côte à côte, le jeu deviendrait alors injouable. Pour pallier à cet inconvénient, nous allons ajouter une variable qui augmente à chaque frame, et qui est remis à zéro lorsque l'on ajoute un trou. Si cette variable est inférieure à 3, on n'ajoute pas de trou mais du sol.

Le code suivant récapitule les ajouts que nous avons réalisés :

```
Génération du terrain
StrRight(Str 1, 20)→Str 1 // on enlève le premier caractère
If RandInt(0,5)=1 And O > 3 // avec un peu d'aléatoire, on ajoute…
Then Str 1+" "→Str 1 // … soit un trou, …
0→O // … et dans ce cas on remet à zéro la distance sans trou ; …
Else Str 1+"T"→Str 1 // … soit du sol
IfEnd
```

Les sauts :

Pour pouvoir gagner, il faut que la boule puisse sauter. Pour savoir si celle-ci est en l'air, nous allons créer une variable qui, selon sa valeur, nous indiquera si la boule est au sol, monte, est en l'air, ou redescend. Nous choisirons la variable « S », comme « saut ». Nous considèrerons la position de la balle en fonction du tableau suivant :

Valeur de S :

- 0 : Au sol
- 1 : Montée
- 2 : En l'air
- 3 : Descente

La boule doit pouvoir sauter que si elle est au sol, sinon elle pourrait rester en l'air, et que si le joueur appuie sur une touche. Nous utiliserons la touche [EXE], mais vous pouvez bien entendu la changer. Aussi, si la boule saute, il faut que sa position à l'écran change. C'est la valeur de la variable P qu'il faudra donc modifier. Nous assignerons à P une nouvelle valeur en fonction de celle de S.

Voici donc un exemple de code qui reprend ce que nous avons vu ci-dessus. Vous remarquerez l'utilité du saut conditionnel « ⇒ » qui permet d'éviter les longues suites de « If… Then… IfEnd ».

```
// Gestion des sauts
Getkey=31 And S=0⇒1→S // si le joueur appuie sur [EXE] et qu'il ne saute pas déjà
S=0⇒4→P // on change la position du joueur en fonction de la durée du saut
S=1 Or S=3⇒3→P
S=2⇒2→P
```

Les collisions :

Pour que le jeu puisse finir, il faut détecter si le joueur est tombé dans un trou. C'est le rôle des collisions, que nous allons approfondir ici.

Premièrement, si la balle est tombée dans un trou, c'est qu'elle ne saute pas, donc que S est égal à zéro. Ensuite, il faut qu'il y ait un trou en dessous d'elle. Pour le savoir, nous allons commencer par découper la chaine de caractère et récupérer le premier caractère, c'est à dire celui qui est en dessous de la boule.

```
StrLeft(Str 1, 1) → Str 2
```

Puis, nous allons le comparer à un trou.

```
If StrCmp(Str 2, " ")=0
Then…
```

En optimisant le tout, on obtient :

```
If StrCmp(StrLeft(Str 1, 1), " ")=0
```

Et en ajoutant la condition si la bille saute ou non :

```
If StrCmp(StrLeft(Str 1, 1), " ")=0 And S>0
```

Puis on ajoute un « Break » ([SHIFT] / [VARS] (PRGM) / [F2] (CTL) / [F3] (Brk)) qui permet de quitter la boucle dans laquelle le programme tourne.

```
If StrCmp(StrLeft(Str 1, 1), " ")=0 And S>0
Then Break
IfEnd
```

L'affichage :

L'affichage d'un jeu est sans doute la partie la plus importante, puisque c'est le rendu à l'écran que nous créons. Sans affichage, le jeu serait injouable, l'écran resterait blanc quelque soit les touches sur lesquelles vous appuyez.

Dans le cas de notre jeu, nous avons le sol, c'est à dire une chaine de caractères, et une boule, un unique caractère à afficher. Nous utiliserons donc les « Locate », la fonction la plus utilisée dans ce genre de programme.

L'affichage de la map sera rapide, puisque la chaine est directement modifiée :

```
Locate 1, 5, Str 1
```

Pour dessiner la boule, nous dessinerons un « o », avec comme position sur l'axe des X la gauche de l'écran, c'est à dire 1, et comme position en Y la valeur de la variable P, celle de la position de la boule.

```
Locate 1, P, "o"
```

Nous afficherons aussi le score tout en bas de l'écran.

```
Locate 1, 7, D
```

Le message de fin de partie :

Pour que le jeu soit un peu plus élaboré, nous pouvons ajouter un petit message lorsque le joueur a perdu. Nous commençons par effacer l'écran avec un « ClrText », puis nous écrivons avec des « Locate ». Nous pouvons aussi y placer le score final.

```
ClrText
Locate 7,3, "Perdu !"
Locate 1,4, "Vous avez parcouru :"
Locate 2,5,D
Locate 7.5. "mètres."
```

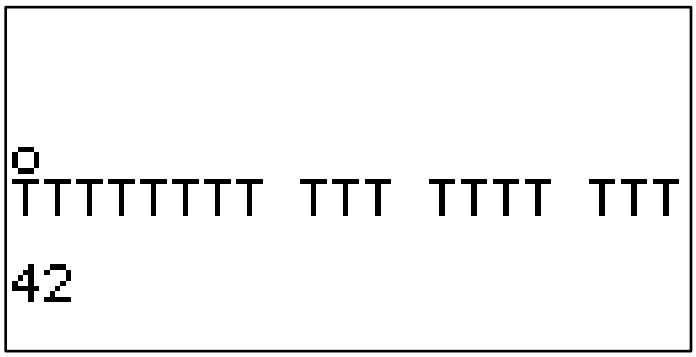
Maintenant que vous avez suivi toutes ces étapes, essayez de créer le jeu vous-même, mais retrouvez en cas de doute l'intégralité du code ci-dessous :

Code du jeu :

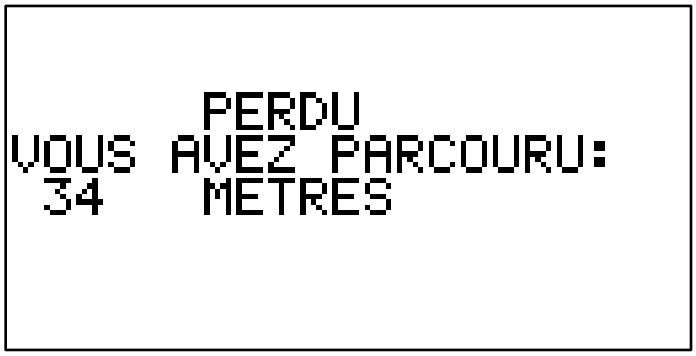
```
// Initialisation
0→S // saut du joueur
0→D // la distance qu'a parcouru le joueur
4→P // la position en Y du joueur
0→O // si un obstacle a récemment été ajouté
0→G // la valeur du Getkey
"TTTTTTTTTTTTTTTTTTTT"→Str 1

While 1
D+1→D // on augmente la distance
O+1→O // on augmente la distance sans trou
S≠0⇒S+1→S // si le joueur saute, on fait progresser le saut
S>3⇒0→S // si il est retombé, on enlève le saut
// Génération du terrain
StrRight(Str 1, 20)→Str 1 // on enlève le premier caractère
If RandInt(0,5)=1 And O > 3 // avec un peu d'aléatoire, on ajoute…
Then Str 1+" "→Str 1 // … soit un trou, …
0→O // … et dans ce cas on remet à zéro la distance sans trou ; …
Else Str 1+"T"→Str 1 // … soit du sol
IfEnd
// Gestion des sauts
Getkey=31 And S=0⇒1→S // si le joueur appuie sur [EXE] et qu'il ne saute pas déjà
S=0⇒4→P // on change la position du joueur en fonction de la durée du saut
S=1 Or S=3⇒3→P
S=2⇒2→P
// Collisions
If StrCmp(StrLeft(Str 1, 1), " ")=0 And S=0 // si on est sur un trou et que l'on ne saute
pas,
Then Break // on quitte la boucle de jeu
IfEnd
// Affichage
ClrText // on efface l'écran
Locate 1,5,Str 1 // on affiche la map
Locate 1,P, "o" // et la boule
Locate 1,7,D // ainsi que la distance

WhileEnd
ClrText // Si on meurt, on affiche un petit mot
Locate 7,3, "Perdu !"
Locate 1,4, "Vous avez parcouru :"
Locate 2,5,D
Locate 7,5, "mètres."
```

Conclusion :

Ce petit jeu vous montre donc la puissance des chaines de caractères, puisque pour faire un jeu de ce type sans, la vitesse serai grandement réduite et la taille du programme beaucoup plus importante.



Quelques images du jeu



Aussi, rien ne vous empêche de l'améliorer, en mettant par exemple un système de meilleurs scores avec une liste, une boucle qui accélère le programme progressivement, ou encore ajouter des obstacles au dessus du sol. Si vous bloquez sur un programme, n'hésitez pas à demander de l'aide sur Planète-Casio (lien ci-dessous), des membres expérimentés tacheront de vous répondre.

A bientôt sur la page Facebook Casio Calculatrices, et sur le site www.planete-casio.fr !