

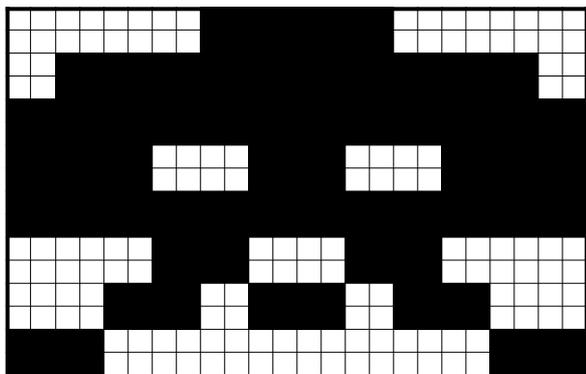
T.P. 8

Space Invaders (partie 11)

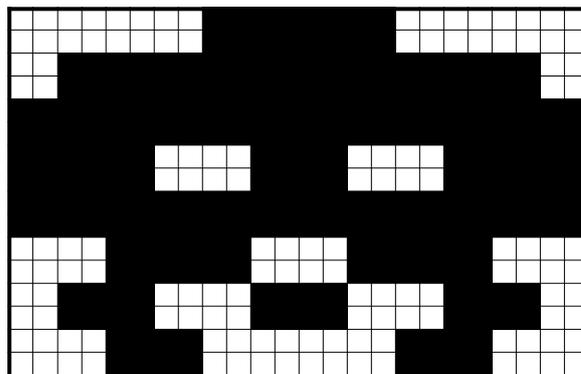
Étape 1

Nous allons ajouter une petite animation à nos envahisseurs. Pour cela, nous utiliserons les couples de bitmaps suivants :

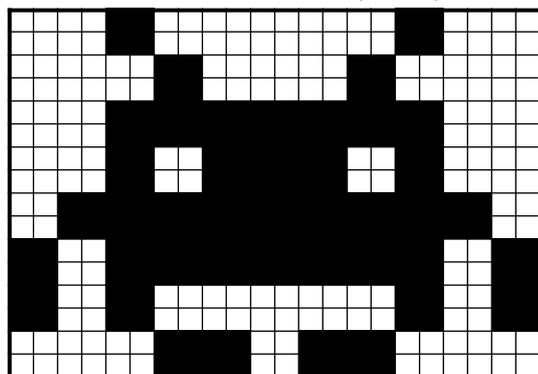
Envahisseur A1 – (24,16)



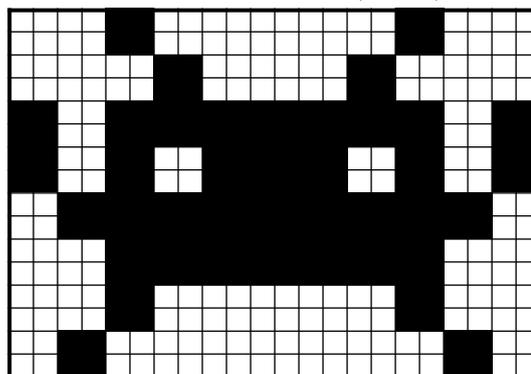
Envahisseur A2 – (24,16)



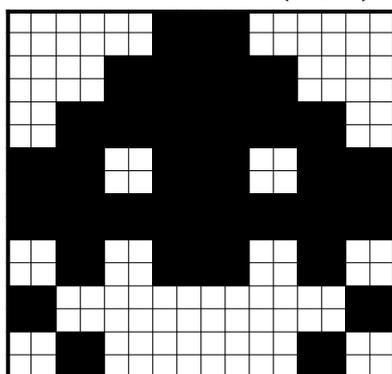
Envahisseur B1 – (22,16)



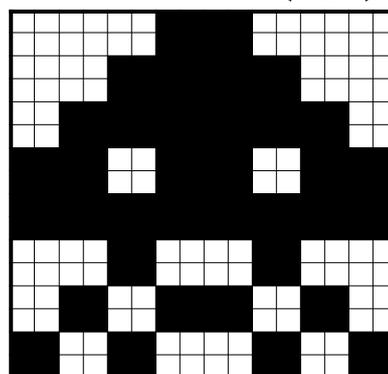
Envahisseur B2 – (22,16)



Envahisseur C1 – (16,16)



Envahisseur C2 – (16,16)



Les envahisseurs A1, B1 et C1 sont ceux que vous utilisez déjà dans votre programme. Commencez donc par remplacer respectivement toutes les occurrences `InvaderA_Bitmap`, `InvaderB_Bitmap` et `InvaderC_Bitmap` par les occurrences `InvaderA1_Bitmap`, `InvaderB1_Bitmap` et `InvaderC1_Bitmap`.

Créez ensuite les bitmaps InvaderA2_Bitmap, InvaderB2_Bitmap et InvaderC2_Bitmap :

InvaderA1_Bitmap	dc.w	24,16
	; ...	
InvaderA2_Bitmap	dc.w	24,16
	dc.b	%00000000,%11111111,%00000000
	dc.b	%00000000,%11111111,%00000000
	dc.b	%00111111,%11111111,%11111100
	dc.b	%00111111,%11111111,%11111100
	dc.b	%11111111,%11111111,%11111111
	dc.b	%11111111,%11111111,%11111111
	dc.b	%11111100,%00111100,%00111111
	dc.b	%11111100,%00111100,%00111111
	dc.b	%11111111,%11111111,%11111111
	dc.b	%11111111,%11111111,%11111111
	dc.b	%00001111,%11000011,%11110000
	dc.b	%00001111,%11000011,%11110000
	dc.b	%00111100,%00111100,%00111100
	dc.b	%00111100,%00111100,%00111100
	dc.b	%00001111,%00000000,%11110000
	dc.b	%00001111,%00000000,%11110000
InvaderB1_Bitmap	dc.w	22,16
	; ...	
InvaderB2_Bitmap	dc.w	22,16
	dc.b	%00001100,%00000000,%11000000
	dc.b	%00001100,%00000000,%11000000
	dc.b	%00000011,%00000011,%00000000
	dc.b	%00000011,%00000011,%00000000
	dc.b	%11001111,%11111111,%11001100
	dc.b	%11001111,%11111111,%11001100
	dc.b	%11001100,%11111100,%11001100
	dc.b	%11001100,%11111100,%11001100
	dc.b	%00111111,%11111111,%11110000
	dc.b	%00111111,%11111111,%11110000
	dc.b	%00001111,%11111111,%11000000
	dc.b	%00001111,%11111111,%11000000
	dc.b	%00001100,%00000000,%11000000
	dc.b	%00001100,%00000000,%11000000
	dc.b	%00110000,%00000000,%00110000
	dc.b	%00110000,%00000000,%00110000
InvaderC1_Bitmap	dc.w	16,16
	; ...	
InvaderC2_Bitmap	dc.w	16,16
	dc.w	%0000001111000000
	dc.w	%0000001111000000
	dc.w	%0000111111110000
	dc.w	%0000111111110000
	dc.w	%0011111111111100
	dc.w	%0011111111111100
	dc.w	%1111001111001111
	dc.w	%1111001111001111
	dc.w	%1111111111111111
	dc.w	%1111111111111111
	dc.w	%0000110000110000
	dc.w	%0000110000110000
	dc.w	%0011001111001100
	dc.w	%0011001111001100
	dc.w	%1100110000110011
	dc.w	%1100110000110011

Modifiez maintenant le sous-programme **InitInvaders** afin d'initialiser les seconds bitmaps pour tous les sprites d'envahisseurs (registre **A2**).

Réalisez le sous-programme **SwapBitmap** qui permute les bitmaps 1 et 2 d'un sprite. Il s'agit simplement d'inverser les valeurs contenues dans les adresses **BITMAP1** et **BITMAP2**.

Entrée : **A1.L** = Adresse du sprite dont on souhaite inverser les bitmaps.

Pour terminer, modifiez le sous-programme **MoveAllInvaders** afin d'appeler **SwapBitmap** pour chaque envahisseur qui vient d'être déplacé.

Relancez votre programme principal et vérifiez que les envahisseurs changent bien de bitmap après chaque déplacement.

Étape 2

Dans cette étape, nous allons détecter la collision entre le tir du vaisseau et les envahisseurs. Un envahisseur touché par un tir sera alors détruit. Nous en profiterons pour décrémenter un compteur qui contiendra le nombre d'envahisseurs encore vivants.

À l'aide de la directive **DC**, réservez un espace mémoire de 16 bits dans la partie « Données » de votre fichier source. Cet espace sera nommé **InvaderCount** et sera utilisé pour contenir le nombre d'envahisseurs encore vivants (cette valeur sera non signée). Vous l'initialiserez avec la constante **INVADER_COUNT** ; c'est-à-dire le nombre maximum d'envahisseurs, car au début du jeu, tous les envahisseurs sont vivants.

InvaderX	dc.w	(VIDEO_WIDTH-(INVADER_PER_LINE*32))/2	; Abscisse globale
InvaderY	dc.w	32	; Ordonnée globale
InvaderCurrentStep	dc.w	INVADER_STEP_X	; Pas en cours
InvaderCount	dc.w	INVADER_COUNT	; Cpt. d'envahisseurs

Réalisez le sous-programme **DestroyInvaders** qui supprime un envahisseur en contact avec le tir du vaisseau (il faudra tester tous les envahisseurs). S'il existe un contact entre un envahisseur et un tir, alors les états respectifs de ces deux derniers seront positionnés à **HIDE** et le compteur **InvaderCount** sera décrémenté de un. Vous testerez votre sous-programme à l'aide du programme principal suivant :

Main	jsr	InitInvaders
\loop	jsr	PrintShip
	jsr	PrintShipShot
	jsr	PrintInvaders
	jsr	BufferToScreen
	jsr	DestroyInvaders
	jsr	MoveShip
	jsr	MoveInvaders
	jsr	MoveShipShot
	jsr	NewShipShot
		bra

Étape 3

Afin d'augmenter la difficulté du jeu, nous allons accélérer le déplacement des envahisseurs au fur et à mesure qu'ils se font détruire. En d'autres mots, moins il y aura d'envahisseurs, plus ils seront rapides.

Lors d'un TP précédent, vous avez réalisé le sous-programme **MoveInvaders** qui appelait une fois sur huit le sous-programme **MoveAllInvaders**. Cela permettait de rendre les envahisseurs huit fois moins rapides que le vaisseau. Afin d'accélérer progressivement le mouvement des envahisseurs, nous allons donc diminuer graduellement la valeur 8 en fonction du nombre d'envahisseurs encore vivants.

La première chose à faire est donc de réserver un espace mémoire qui contiendra la valeur à décrémenter. Utilisez pour cela la directive DC. L'espace mémoire sera de 16 bits, s'appellera **InvaderSpeed** et sera initialisé à la valeur 8 (on considérera cette valeur comme non signée).

InvaderX	dc.w	(VIDEO_WIDTH-(INVADER_PER_LINE*32))/2	; Abscisse globale
InvaderY	dc.w	32	; Ordonnée globale
InvaderCurrentStep	dc.w	INVADER_STEP_X	; Pas en cours
InvaderCount	dc.w	INVADER_COUNT	; Cpt. d'envahisseurs
InvaderSpeed	dc.w	8	; Vitesse (1 -> 8)

Remplacez ensuite la valeur immédiate #8 dans **MoveInvaders** par un adressage absolu long qui pointerait vers **InvaderSpeed**.

Il nous faut maintenant définir la vitesse des envahisseurs en fonction de leur nombre. C'est-à-dire définir la valeur d'**InvaderSpeed** en fonction de celle d'**InvaderCount**. Pour cela, nous respecterons les valeurs du tableau ci-dessous :

InvaderCount	InvaderSpeed
1	1 (vitesse maximale)
De 2 à 5	2
De 6 à 10	3
De 11 à 15	4
De 16 à 20	5
De 21 à 25	6
De 26 à 35	7
De 35 à 50	8 (vitesse minimale)

À partir des valeurs du ci-dessus, nous allons créer un tableau contenant les différents paliers de vitesse à franchir pour incrémenter **InvaderSpeed** :

InvaderX	dc.w	(VIDEO_WIDTH-(INVADER_PER_LINE*32))/2	; Abscisse globale
InvaderY	dc.w	32	; Ordonnée globale
InvaderCurrentStep	dc.w	INVADER_STEP_X	; Pas en cours
InvaderCount	dc.w	INVADER_COUNT	; Cpt. d'envahisseurs
InvaderSpeed	dc.w	8	; Vitesse (1 -> 8)
SpeedLevels	dc.w	1,5,10,15,20,25,35,50	; Paliers de vitesse

Réalisez maintenant le sous-programme **SpeedInvaderUp** qui modifie le contenu d'InvaderSpeed en fonction du nombre d'envahisseurs encore vivants. Utilisez le tableau SpeedLevels pour positionner la valeur d'InvaderSpeed.

Indications :

- Réalisez une boucle qui va comparer le nombre d'envahisseurs aux différents paliers contenus dans SpeedLevels (chaque itération de la boucle compare le nombre d'envahisseurs à un palier).
- Le contenu d'InvaderSpeed est incrémenté à chaque itération de la boucle.
- La boucle s'arrête quand le nombre d'envahisseurs est supérieur au palier.

Testez votre sous-programme à l'aide du programme principal suivant :

```
Main      jsr    InitInvaders
\loop     jsr    PrintShip
          jsr    PrintShipShot
          jsr    PrintInvaders
          jsr    BufferToScreen

          jsr    DestroyInvaders

          jsr    MoveShip
          jsr    MoveInvaders
          jsr    MoveShipShot

          jsr    NewShipShot
          jsr    SpeedInvaderUp

          bra   \loop
```