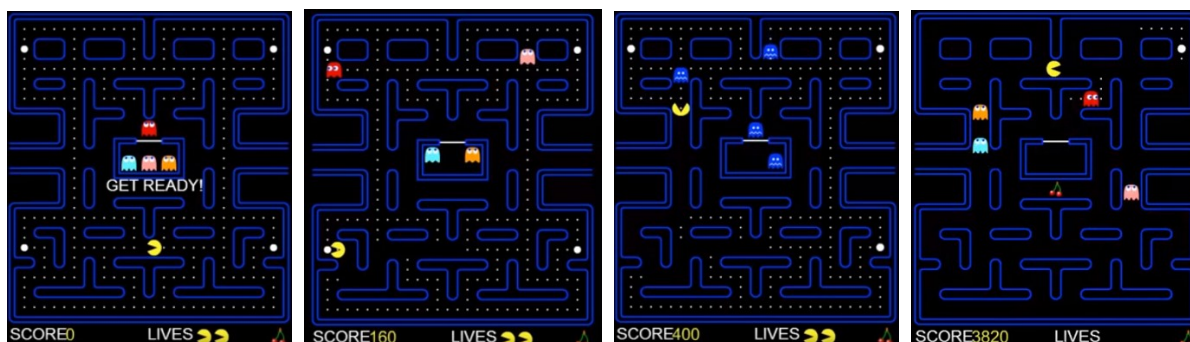


PAC-MAN


En mai 1980, la première borne d'arcade Pac-Man était installée au Japon. Le héros en forme de rond jaune se déplace à l'intérieur d'un labyrinthe. Il doit ramasser l'ensemble des pac-gommes pour passer au niveau suivant. Cependant, pour corser l'affaire, quatre fantômes sont à sa poursuite : Shadow en rouge, Speedy en rose, Bashful en bleu et Pokey en orange. Vous pouvez trouver étrange que l'on baptise des fantômes, mais au Japon, cela est possible ! Si un des fantômes réussit à rattraper Pacman, c'en est fini pour lui. Mais il y a une justice en ce bas monde. Quatre supers pac-gommes sont réparties aux quatre coins du labyrinthe. Lorsque Pacman en avale une, c'est le monde à l'envers : il peut enfin dévorer les fantômes, qui ayant compris ce qu'il se passe, se défilent à la vitesse grand V. Bien sûr, cet effet reste de courte durée et Pacman devra rapidement redevenir un glouton surfant entre ses ennemis.



Le jeu a connu un succès immense et a été décliné dans différentes versions sur consoles et bornes d'arcade. Il fut une époque où chaque mois sortait un clone du jeu sur ordinateur, on parlait alors du Pac-Man du mois. Et oui, c'était l'époque de la Pacmania ! Quelques liens pour présenter le jeu :

- PACMAN l'original : <https://www.youtube.com/watch?v=-CbyAk3Sn9I>
- <https://www.youtube.com/watch?v=XQicy0CfQuo>
- **ANDROID** : <https://play.google.com/store/apps/details?id=com.namcobandaigames.pacmantournaments>
- **APPLE** : <https://apps.apple.com/fr/app/id293778748>

Au centre du décor se trouve une sorte de maison où viennent ressusciter les fantômes dévorés. Une petite porte leur permet de sortir. Mais Pac-man n'a pas accès à cette zone.

Durant la partie, des bonus peuvent être ramassés comme par exemple  la cerise, la fraise, la pastèque, la pomme ou le raisin... Ils n'ont pas d'intérêt particulier dans le Gameplay sinon vous rapporter des points en plus et vous forcer à vous rapprocher du centre du décors.

Pour simplifier le codage de l'IA, nous allons mettre en place une version tour par tour. A chaque itération, les fantômes ainsi que Pacman vont se déplacer d'une case. C'est un prérequis pour garder une animation fluide. Effectivement, dans le jeu original, les fantômes étaient un peu plus rapides que Pacman. Mais rassurez-vous, cela ne vous empêchera pas d'avoir des sueurs froides avec cette version simplifiée !

Etape 1 : prendre en main le programme d'exemple

- Chargez et exécutez le programme PACMAN.py.
- Faites en sorte que Pacman mange les pac-gommes !
- Faites évoluer le score en fonction.

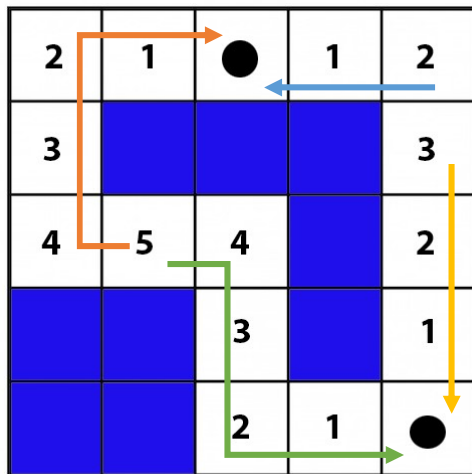
Etape 2 : donner une IA à PAC-MAN

Pour l'instant, la présence des fantômes est peu critique. L'objectif de Pacman est centré uniquement sur avaler l'ensemble des pac-gommes du décor. Ce n'est pas chose simple ! Pour cela, nous vous proposons de construire une carte des distances. Voici un exemple dans le schéma ci-dessous. Il y a deux pac-gommes dans le décor. Nous ne faisons pas de différence entre les pac-gommes standards et les super pac-gommes à ce niveau. Les murs sont dessinés en bleu. Ce qui nous intéresse, ce sont les cases vides. Normalement, Pac-man est toujours disposé sur une case vide car il vient de dévorer la pac-gomme de la case où il est arrivé.

2	1	●	1	2
3				3
4	5	4		2
		3		1
		2	1	●

L'idée à la fin de l'algorithme est que chaque case corresponde à la distance (en cases parcourues) à la pac-gomme la plus proche. Ainsi, lorsque l'on regarde la case du rond rouge, elle porte le chiffre 2 car si Pac-man se trouvait sur cette case, il serait exactement à deux cases de la pac-gomme de droite. La case avec un rond-vert contient le chiffre 5. Chose amusante, elle se situe à égale distance des deux pac-gommes. Ainsi que l'on parte à droite, ou à gauche, il faudra parcourir 5 cases à Pac-man pour arriver à destination. Lorsque Pac-man est sur une case, comment sait-il où aller ? En fait, s'il cherche à se diriger vers la pac-gomme la plus proche, il suffit qu'il recherche sur les 4 cases avoisinantes (en haut, en bas, à gauche, à droite), la case contenant une valeur plus faible. Cette case

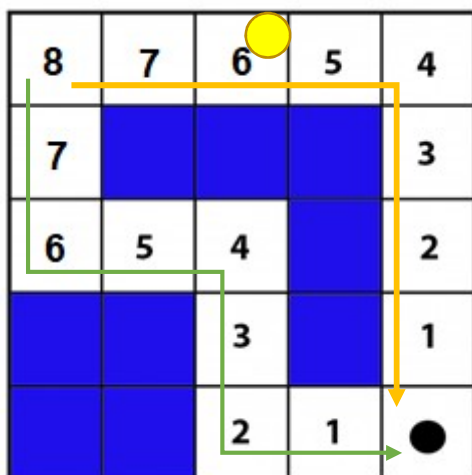
existe toujours et elle indique le chemin vers la destination. Voici en exemple le tracé des chemins suivant cette règle :



Ainsi, si Pac-man se trouve en haut à droite, il aura tendance à se déplacer vers la gauche pour attraper la pac-gomme du haut. S'il se trouve sur la colonne de gauche, il a intérêt à aller vers le haut pour attraper la même pac-gomme.

Mais, comment Pacman sait-il où aller après avoir mangé la pac-gomme en haut du décor, les cases avoisinantes lui indiquant de se diriger vers cette case. Une fois que la pac-gomme a été dévoré, la carte des distances change. En effet, comme il n'y a plus de pac-gomme dans ce secteur, les valeurs évoluent et

maintenant nous obtenons le tracé suivant :



Ainsi, lorsque Pac-man vient de dévorer la pac-gomme en haut, les valeurs de la carte des distances évoluent dans la foulée. Maintenant, il doit pour se diriger le plus rapidement possible vers la nouvelle pac-gomme suivre le parcours en orange.

Au début de la partie, cet algorithme a peu d'intérêt car il y a des pac-gommes partout. Ainsi, quelle que soit la direction choisie par Pac-man, tout va bien se passer. Une méthode qui choisirait une direction au hasard à chaque croisement serait tout aussi efficace.

Cependant, une fois que 50%, 80% des pac-gommes auront été mangées, les choses deviennent moins évidentes. Et lorsqu'il ne reste plus qu'une seule pac-gomme dans le décor, si vous n'utilisez pas cette méthode ou une méthode équivalente, Pac-man risque de tourner en rond pendant un bon moment !

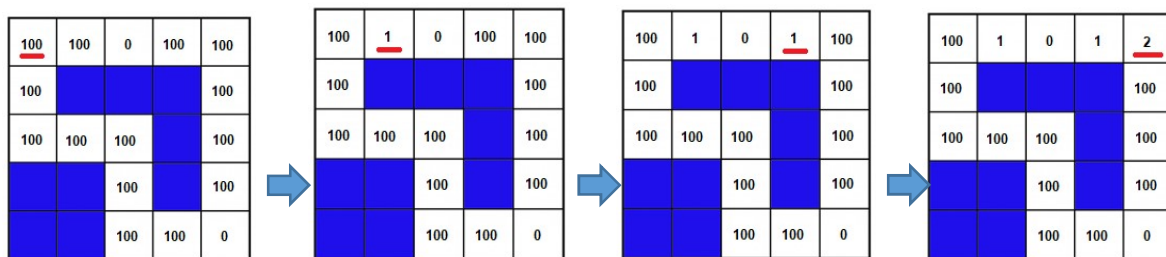
Il nous reste maintenant à présenter comment construire cette carte des distances. Nous allons en premier lieu créer une grille de la même taille que la grille de jeu. Les cases correspondantes aux murs seront initialisées avec une valeur infiniment grande. Les cases du parcours seront initialisées à une valeur correspondant au plus long parcours possible dans le décor, soit par exemple 100. Ensuite, les cases correspondantes à des pac-gommes seront initialisées à la valeur 0.

Par la suite, nous allons parcourir l'ensemble des cases. Si une case correspond à un mur, nous l'ignorons. Si une case porte une valeur positive, alors nous prenons les valeurs des 4 cases avoisinantes. Nous calculons la valeur minimale de ces valeurs et nous mettons à jour la valeur de la case courante avec cette valeur minimale + 1 **uniquement si cela diminue la valeur de la case courante**. La logique est la suivante :

- Si je suis sur une case de valeur 0 entourée par deux cases de valeurs 100, la valeur de la case courante ne passe pas à 101. Idem pour une case de valeur 100 entourée de plusieurs cases à la valeur 100, sa valeur ne passe pas à 101 car cela augmenterait sa valeur actuelle.
- Si la case à ma gauche est à 3 cases d'une pac-gomme, que la case à ma droite est à 6 cases d'une autre pac-gomme et que les cases en haut et en bas sont des murs, alors j'en conclus que la pac gomme la plus-proche de ma case se situe à $3 + 1$ cases.

Pour traiter la grille, nous allons parcourir les cases de gauche à droite puis de haut en bas.

La valeur contenue dans le coin en haut à gauche ne change pas car les valeurs des cases voisines sont à 100. La deuxième case en partant de la gauche porte la valeur 100 et elle est entourée par les valeurs 0 et 100, elle passe donc à la valeur 1. La troisième case vaut 0, sa valeur ne peut changer. La 4^{ème} case vaut 100 et elle est entourée par les valeurs 0 et 100, sa valeur passe donc à 1. La dernière case de cette ligne porte la valeur 100, elle est entourée par les valeurs 1 et 100, sa valeur passe donc à 2.



Nous continuons à la deuxième ligne. La valeur 100 reste inchangée car les deux cases avoisinantes sont à la valeur 100. La case tout à droite est entourée par les valeurs 2 et 100, sa valeur passe à 3. Comme nous parcourons le tableau de gauche à droite et de haut en bas, les modifications ont tendance à se reporter plus facilement dans ces directions. La prochaine case modifiée est celle sous la valeur 3 qui passe alors à la valeur 4. La case se trouvant sous la valeur 4, se trouve maintenant entourée par les valeurs 0 et 4. Ainsi, d'après nos règles de transformation, sa valeur passe à 1. La dernière valeur modifiée est celle de la case à gauche de la valeur 0. Elle est entourée par les valeurs 0 et 100, elle passe donc à la valeur 1. Il s'agit de la dernière mise à jour effectuée lors de ce premier parcours de grille.



Il est évident que ce calcul n'est pas terminé. Nous allons donc relancer un nouveau parcours.

Cette fois la case en haut à gauche est mise à jour pour la valeur 2. Il n'y aura pas d'autres modifications sur cette ligne. La case en dessous passe ainsi à la valeur 3. La valeur 3 en fin de ligne reste inchangée. La case en dessous du 3 dans la colonne de gauche passe à la valeur 4 et les cases immédiatement à sa droite se modifient pour les valeurs 5 et 6 :

2	1	0	1	2
100				3
100	100	100		4
		100		1
		100	1	0

→

2	1	0	1	2
3				3
100	100	100		4
		100		1
		100	1	0

→

2	1	0	1	2
3				3
4	100	100		4
		100		1
		100	1	0

→

2	1	0	1	2
3				3
4	5	6		4
		100		1
		100	1	0

La valeur 4 toute à droite de cette ligne est entourée par un 3 et par un 1. Sa valeur est donc mise à jour pour la valeur 2. A la ligne suivante, la valeur 100 est mise à jour pour la valeur 7. La valeur 1 en fin de ligne ne change pas. A la dernière ligne, la valeur 100 entourée par les valeurs 7 et 1 est alors mise à jour pour la valeur 2.

2	1	0	1	2
3				3
4	5	6		2
		100		1
		100	1	0

→

2	1	0	1	2
3				3
4	5	6		2
		7		1
		100	1	0

→

2	1	0	1	2
3				3
4	5	6		2
		7		1
		2	1	0

La deuxième passe est terminée. Depuis le début du traitement, toutes les cases ont été mises à jour. Pourtant, avons-nous terminé ? Rappelez-vous de la case sur la colonne de droite à la valeur 4, qui finalement a été modifiée une deuxième fois pour prendre la valeur 2 à cause d'une valeur inférieure arrivant par le bas. Ainsi, le tableau est en évolution permanente et des cases peuvent être mises à jour plusieurs fois tant que des meilleurs trajets se forment. Dans la grille actuelle, par exemple, la case portant la valeur 7 n'est pas optimale car située à côté d'une case de valeur 2. Cette valeur 2 va se propager pour créer un 3 et puis un 4. Mais alors, comment savoir à quel moment le traitement s'arrête ?

Critère d'arrêt : lors du traitement complet d'une grille, si aucune case n'est mise à jour, alors les valeurs ont fini de converger, elles correspondent aux valeurs de la carte des distances.

2	1	0	1	2
3				3
4	5	6		2
		3		1
		2	1	0

Nous enclenchons donc notre 3^{ème} parcours de grille.

La valeur 7 est la seule valeur mise à jour. La case correspondante prend alors la valeur 3.

Suivant le critère d'arrêt énoncé, une modification a été effectuée, nous relançons donc le processus.

2	1	0	1	2
3				3
4	5	4		2
		3		1
		2	1	0

Nous enclenchons donc notre 4^{ème} parcours de grille.

La valeur 6 est la seule valeur mise à jour pour la valeur 4.

Suivant le critère d'arrêt énoncé, une modification a été effectuée, nous relançons le processus. Cette fois, le parcours complet ne donne lieu à aucune modification. Ce parcours, qui peut sembler inutile, sert à nous confirmer que nous avons convergé vers la solution. Les valeurs trouvées correspondent à la carte des distances.

Etape 3 : fantômes

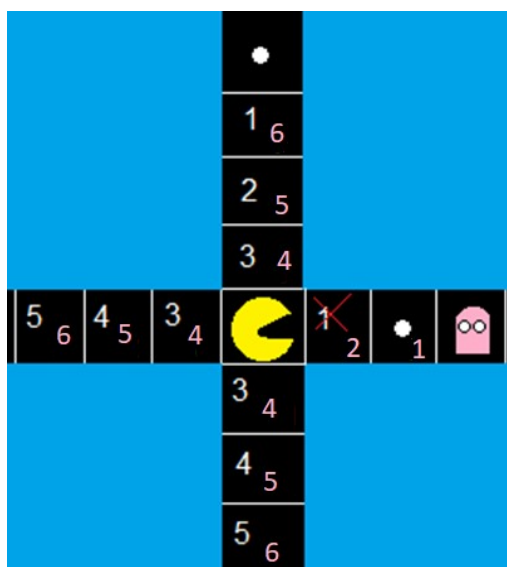
Nous allons donner une IA de chasse assez basique aux fantômes. Chaque fantôme va avoir une direction courante : haut, bas, gauche, droite. Dans un couloir, chaque fantôme avance dans sa direction courante. Cependant, dès lors qu'il arrive à un changement de direction ou à un croisement, les différentes directions possibles sont analysées et une nouvelle direction est tirée au hasard.

Petit conseil, pour détecter la configuration du couloir, il faut que les directions possibles correspondent exactement à la direction courante et à son opposée.

Programmez la collision entre un fantôme et Pacman. Même si les IA des fantômes sont faibles, ils sont assez nombreux pour lui barrer la route et l'obliger à réagir.

Etape 4 : éviter les fantômes

Comme pour les pac-gommes, nous allons appliquer la même logique pour calculer une carte des distances de proximité des fantômes. Pour cela, dans un nouveau tableau, nous initialisons notre calcul en mettant à zéro les cases où se trouvent les fantômes et nous construisons itérativement la carte des distances. Quelle est la signification des valeurs contenues dans cette carte ?



Examinez les cases autour de Pacman, si elles contiennent une valeur inférieure ou égale à 3 dans la carte des distances des fantômes, nous considérons qu'un fantôme est trop proche dans cette direction et il faut alors l'ignorer.

Voici un schéma résumant la méthode :

La carte de distance du fantôme rose est indiquée par des chiffres en rose.

La carte des distances des pac-gommes est en blanc. La pac-gomme sur la droite génère une case de valeur 1 à droite de Pacman ce qui devrait l'amener à prendre la décision d'aller vers la droite. Cependant, cette case étant à une distance de 2 du fantôme rose, elle ne doit

pas être retenue. La case à droite de Pacman est donc retirée des cases possibles. Elle ne sera pas un choix de déplacement pour Pacman. Les trois autres cases autour de lui portent la valeur 3. Même si ces valeurs sont maintenant « fausses » car la pac-gomme de droite ne nous intéresse plus, elles constituent un chemin possible de fuite pour notre héros. Et quand il s'agit de survie ! Toute option est bonne à prendre !

Dans le cas extrême où Pacman est entouré par des fantômes et qu'aucune des directions ne peut être retenues, on choisira alors un déplacement au hasard.