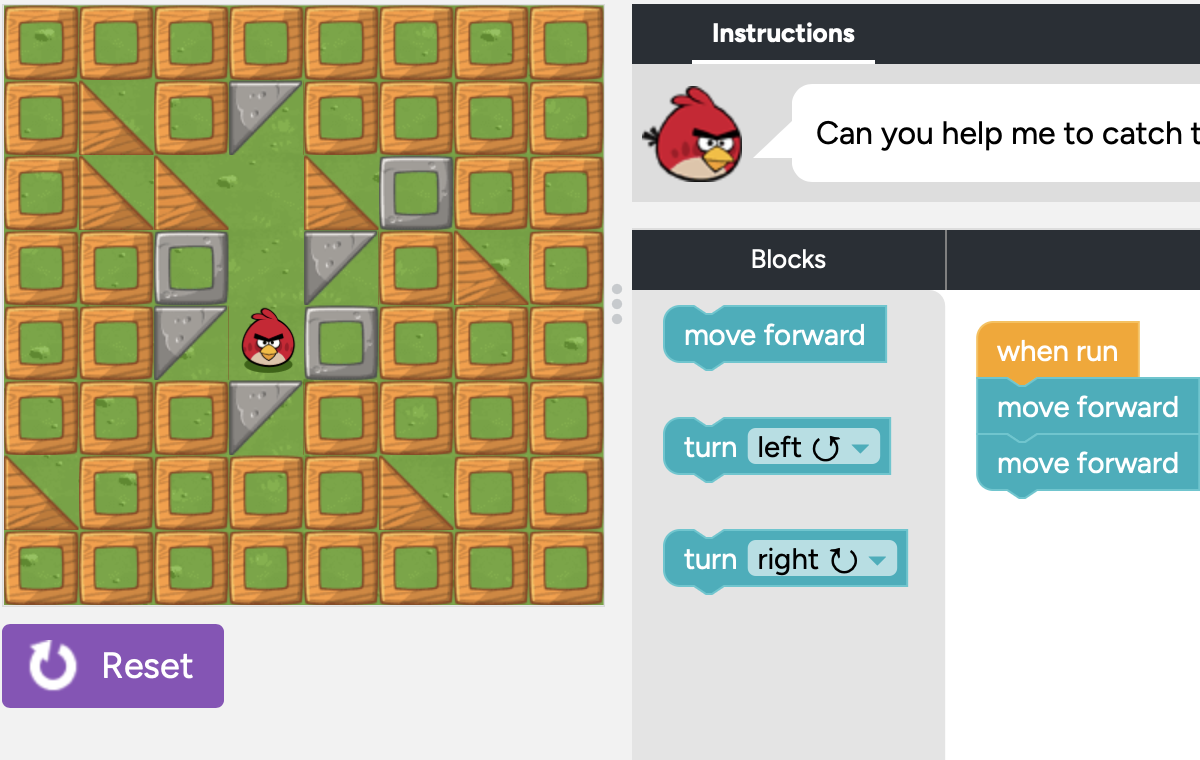
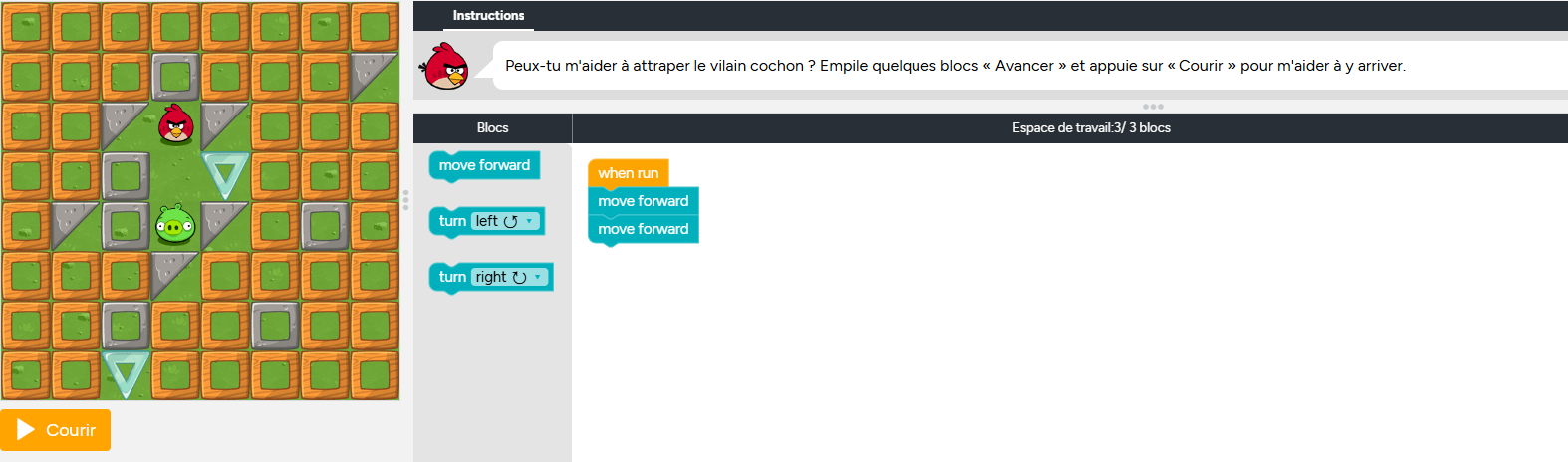
Compte rendu de TP

Matisse Kra  
BTS SIO 1

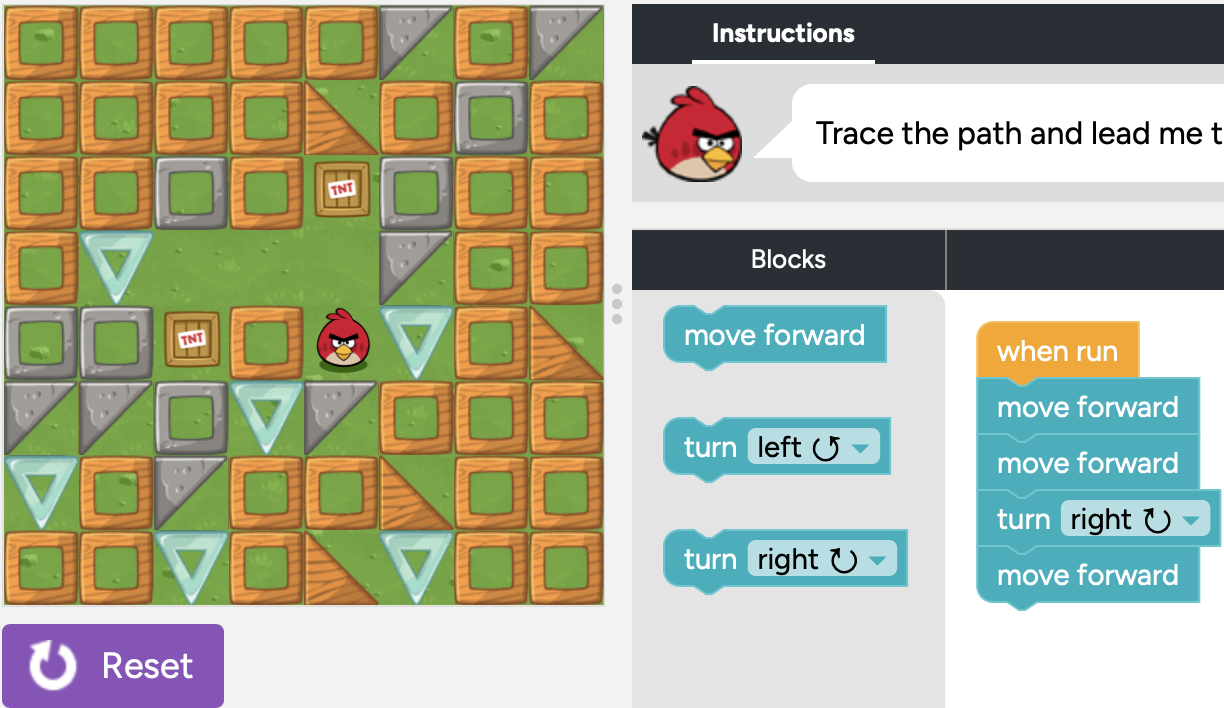
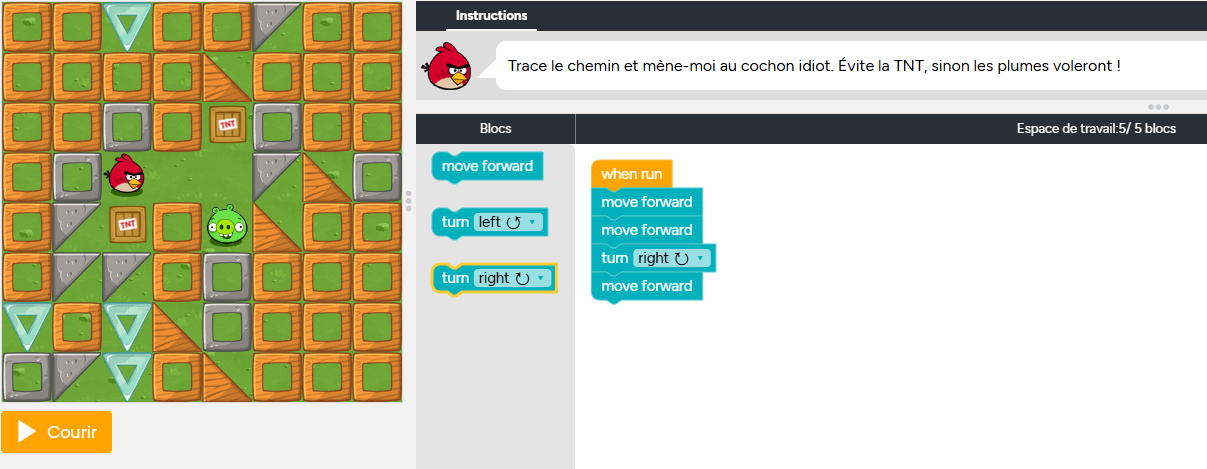
L’objectif principal de ce TP était de concevoir un algorithme permettant à un personnage (l’oiseau ou le zombie) de se déplacer d’un point A à un point B afin d’atteindre une cible (le cochon vert ou le tournesol).  
  
L’activité visait à mettre en pratique les notions de base en algorithmique grâce à l’utilisation de blocs mis à disposition, tout en respectant les contraintes imposées.  
  
Le TP a été réalisé sur les ordinateurs du lycée, sous Windows. Il comporte différents niveaux progressifs (du niveau 1 au niveau 20).

## Niveau 1

L’objectif était de faire parvenir l’oiseau jusqu’au cochon vert.  
- Solution : utilisation de deux blocs “Move forward” pour avancer de 2 cases.  
- Justification : comme le cochon se trouvait en ligne droite, il n’était pas nécessaire d’utiliser les blocs “Turn right” ou “Turn left”.

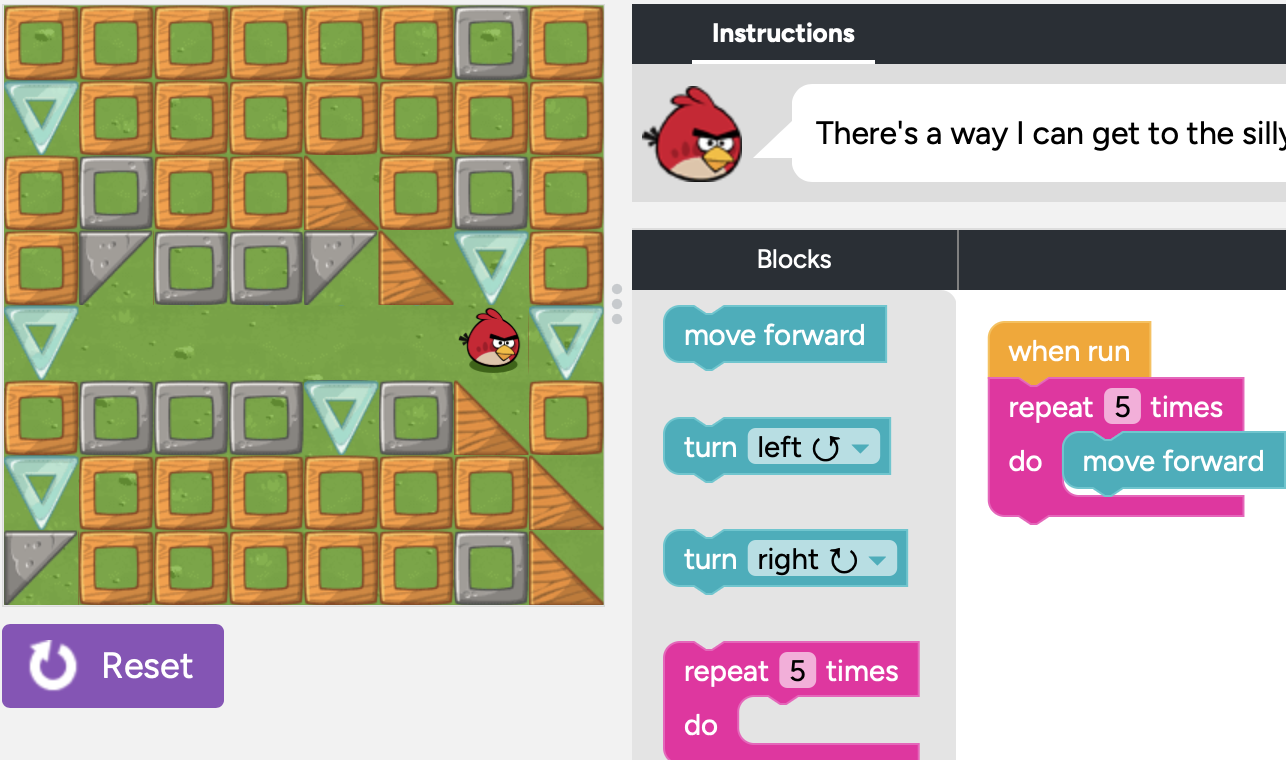


## Niveau 3

Le cochon se trouvait 2 cases en avant et 1 case vers le bas.  
- Solution : introduction du bloc “Turn right” combiné avec “Move forward”.  
- Résultat : l’oiseau a pu descendre et atteindre le cochon.

## Utilisation des boucles (exemple niveau 6)

Certains niveaux demandent d’utiliser des boucles pour éviter les répétitions.  
- Exemple : l’oiseau était à 5 cases du cochon.  
- Sans boucle : empiler 5 fois le bloc “Move forward”.  
- Avec boucle : utilisation du bloc “Repeat 5 times” contenant “Move forward”.  
- Résultat : l’oiseau a avancé de 5 cases directement.



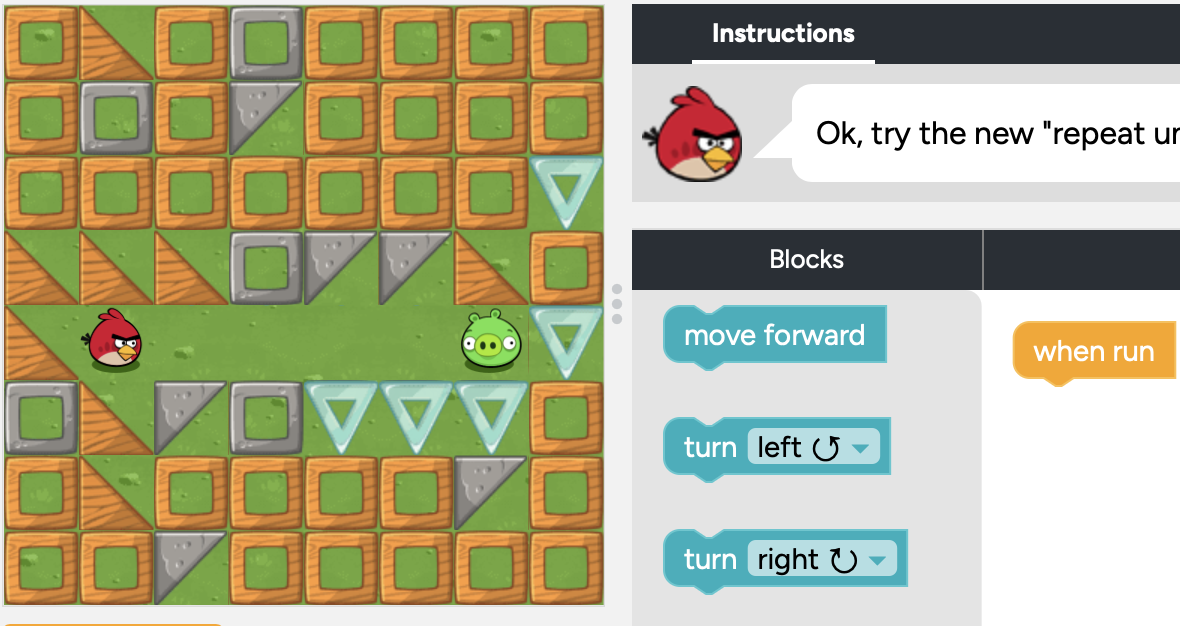
## Contraintes imposées (niveau 9)

Dans ce niveau, un bloc gris était imposé et ne pouvait pas être supprimé.  
- Solution :  
 1. Avancer de 2 cases.  
 2. Tourner à droite.  
 3. Avancer de 2 cases.  
 4. Tourner à droite.  
 5. Avancer pour atteindre le cochon.



## Boucle « Répéter jusqu’à »

Dans certains niveaux, il fallait utiliser la boucle “Repeat until”.  
- Exemple 1 : l’oiseau devait avancer jusqu’à ce qu’il atteigne le cochon.  
- Solution : bloc “Repeat until (cochon atteint)” contenant “Move forward”.  
- Résultat : l’oiseau avançait automatiquement jusqu’à atteindre le cochon.





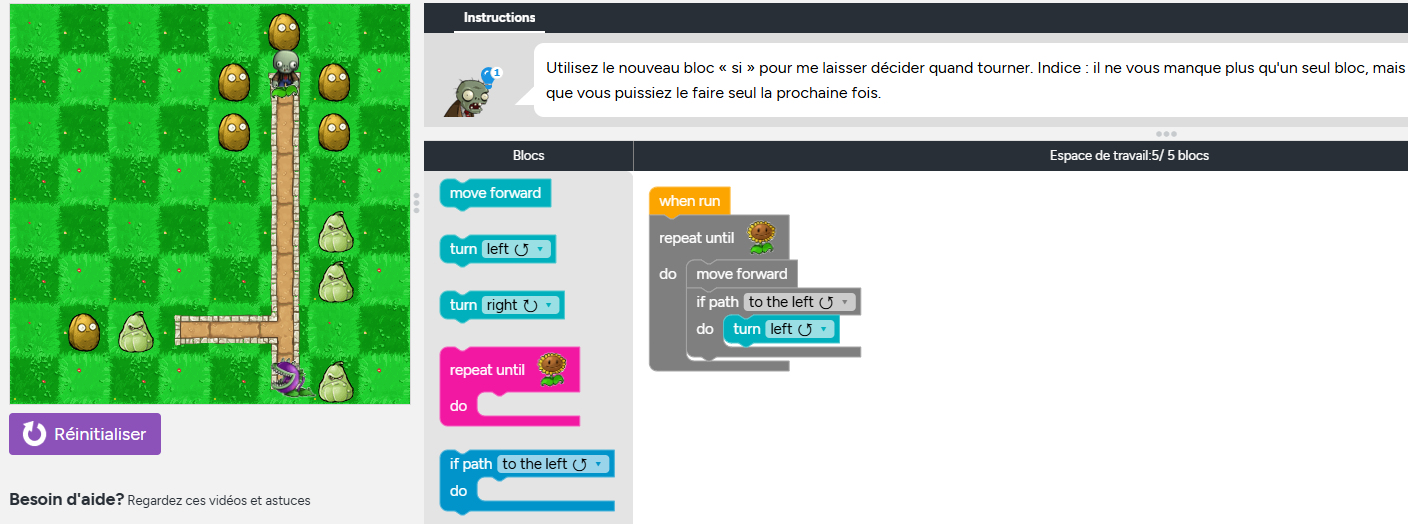
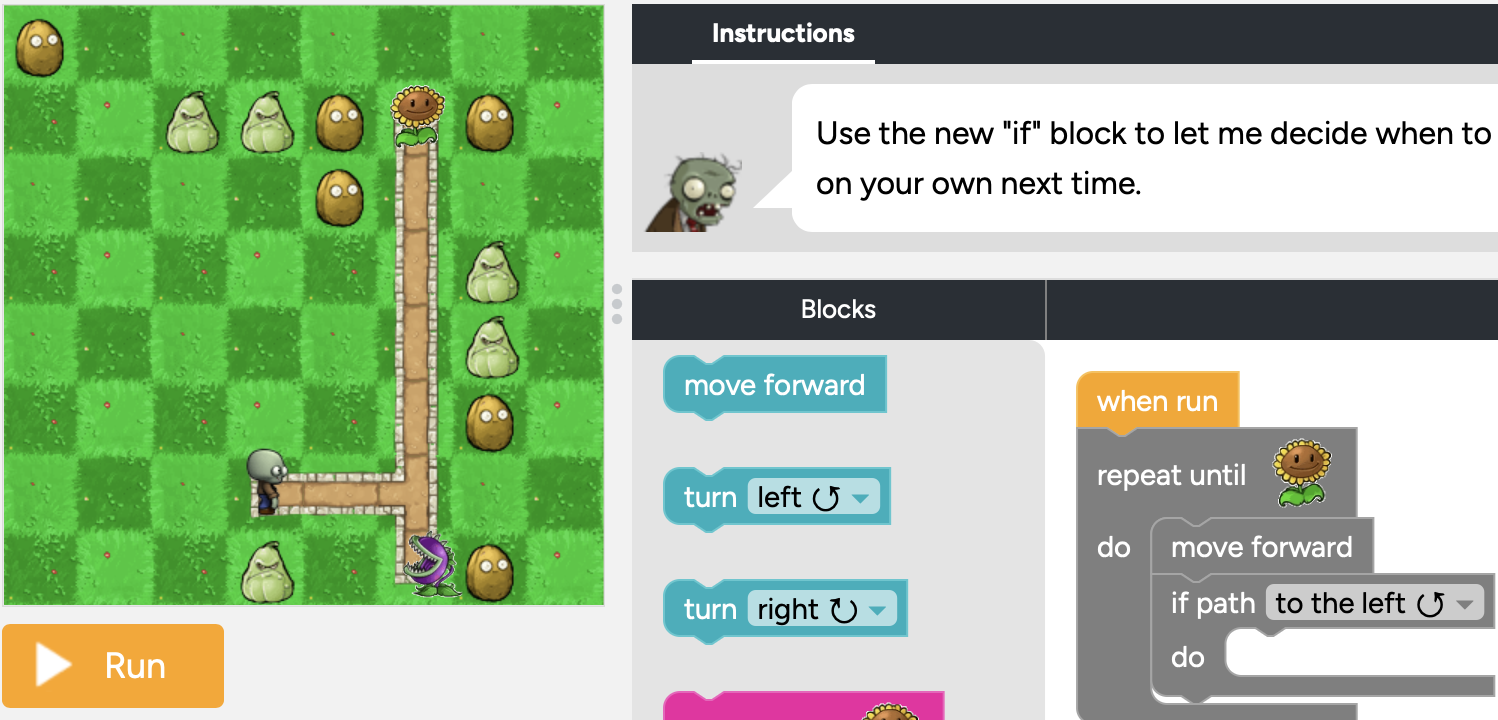
## Exemple 2: le zombie devait atteindre le tourne sol

- Solution : bloc “Repeat until (tourne sol atteint)” contenant “Move forward””turn left et “turn right”.

- Résultat : le zombie avançait et tourner automatiquement jusqu’à atteindre le tourne sol

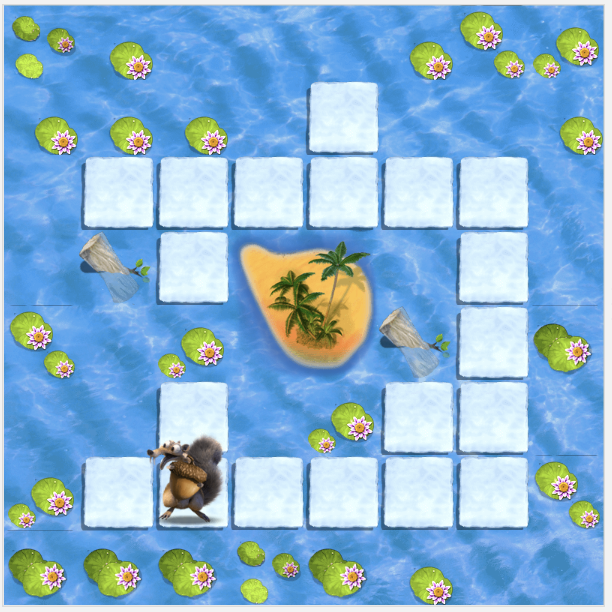
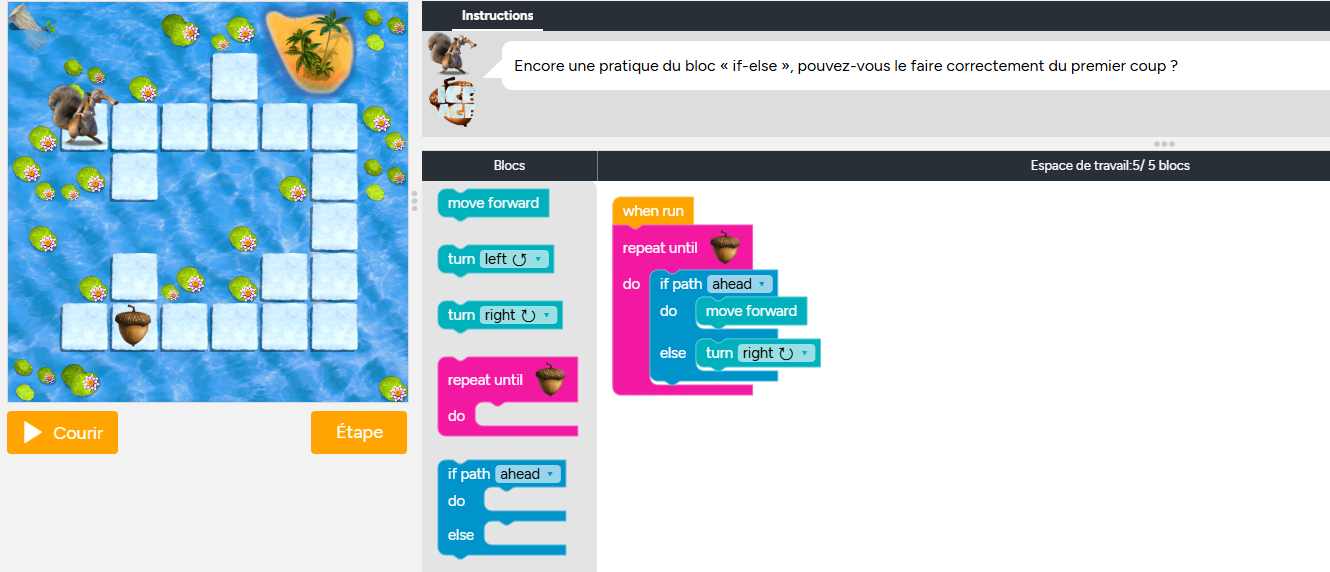


## Niveau 14

Ce niveau imposait l’utilisation du bloc “Repeat until” et du bloc conditionnel “If path then”.  
- Ce bloc permet de donner une certaine autonomie à l’algorithme.  
- Solution : ajout du bloc “Turn left” pour permettre au zombie d’atteindre le tournesol.  


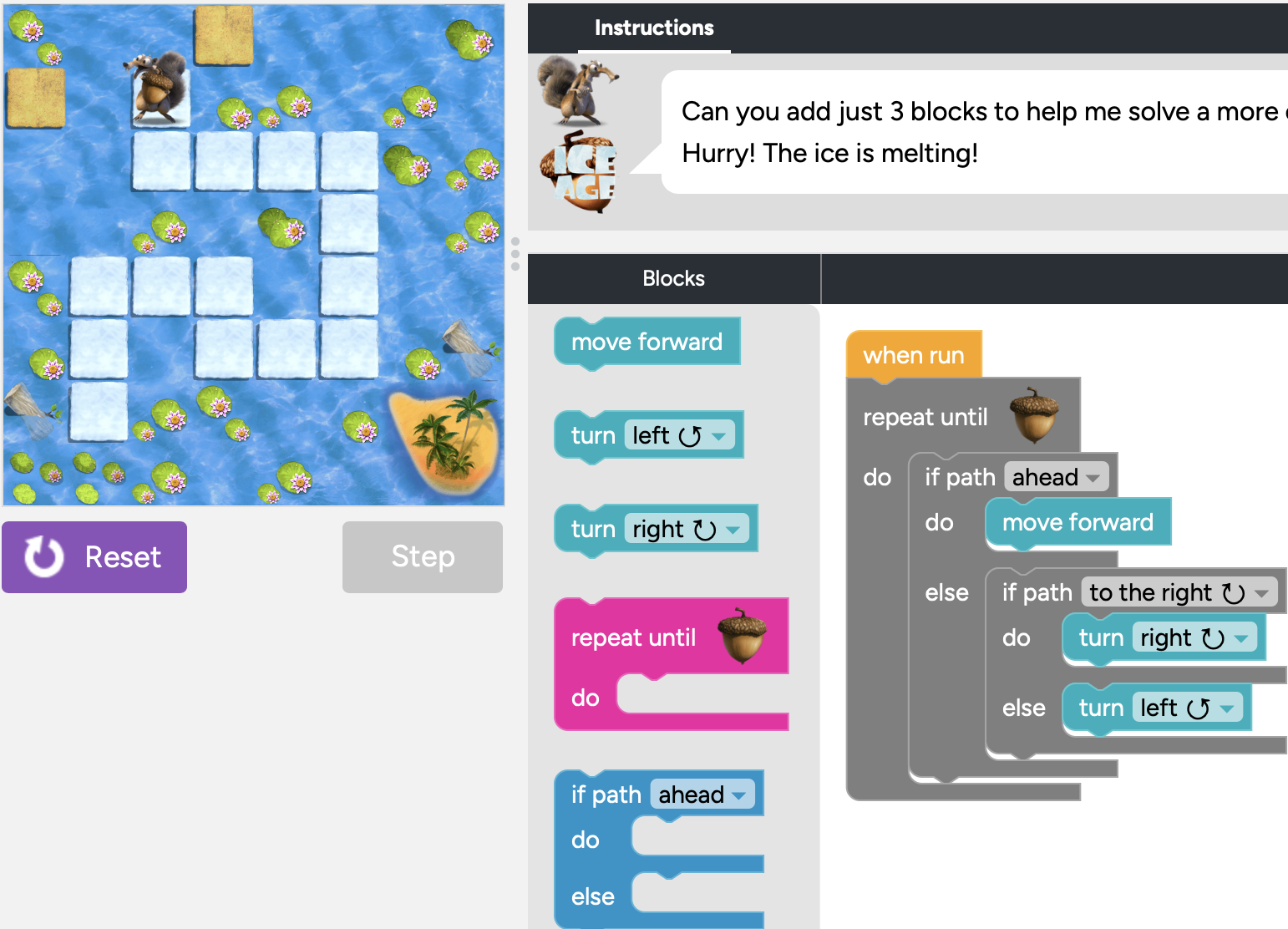
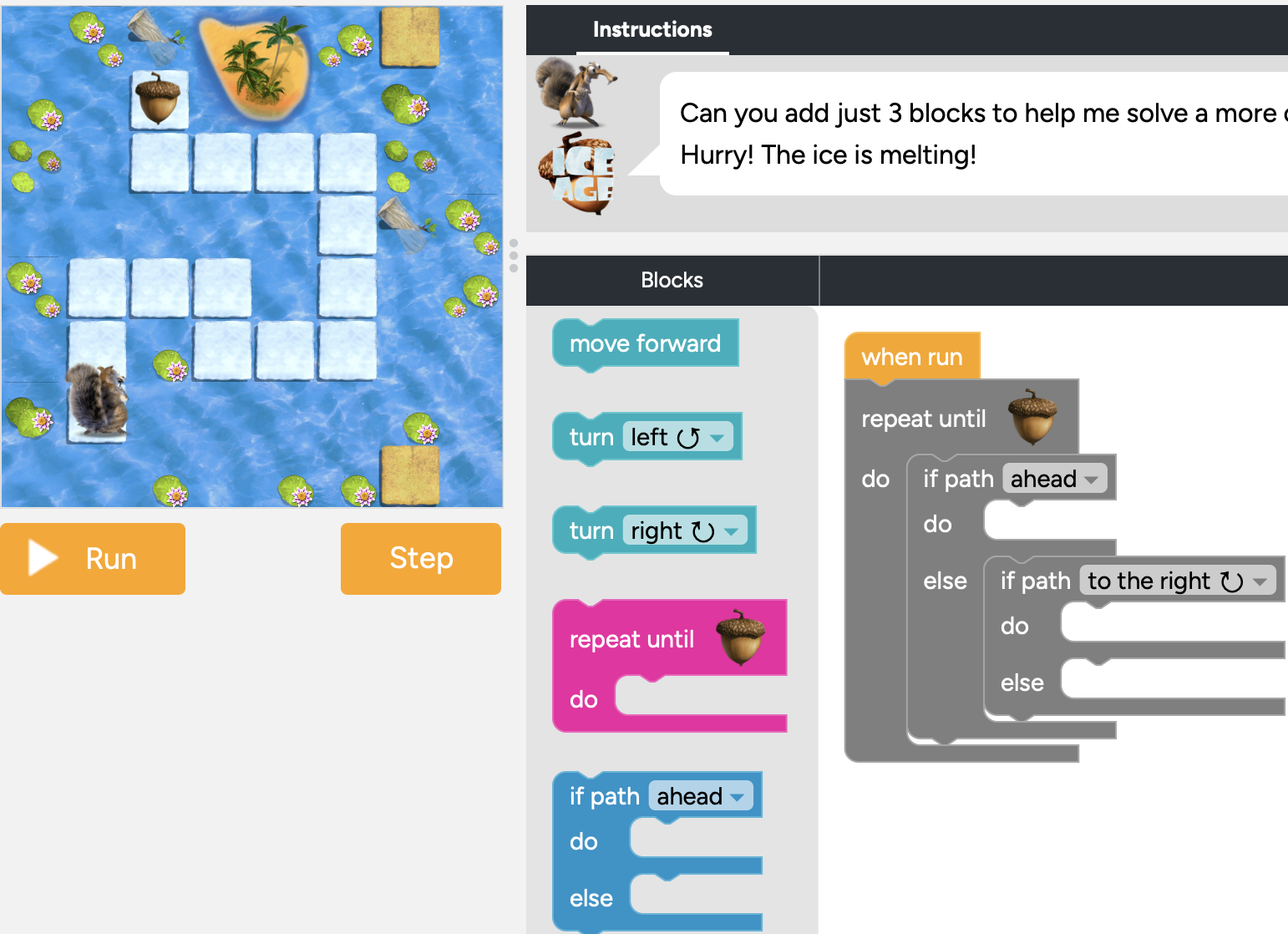
**Niveau 19**

Dans ce niveau, le bloc “If path then… else” a été introduit.  
- Ce bloc permet à l’ordinateur de prendre une décision entre deux options.  
- Résultat : l’algorithme devient plus flexible et peut s’adapter selon les chemins disponibles.



**Niveau 20**

Dans ce niveau il y a une imbrication de bloc répétition et conditionnel avec l’obligation de les utiliser -la combinaison de ses bloc permet au programme d’être parfaitement autonome -solution : on ajoute les bloc de mouvement, avancer, tourner à gauche,tourner a droite



# 3. Résultats et analyse

Ce TP m’a permis de comprendre l’importance de la structuration des algorithmes à travers l’utilisation de blocs visuels.  
  
J’ai revus:  
- Les actions de base (“Move forward”, “Turn left”, “Turn right”).  
- Les boucles fixes (“Repeat n times”).  
- Les boucles conditionnelles (“Repeat until”).  
- Les instructions conditionnelles (“If path then”, “If path then… else”).  
  
Les contraintes imposées m’ont obligé à réfléchir à différentes solutions, ce qui a renforcé ma logique algorithmique et ma capacité à résoudre des problèmes.

# 4. Conclusion

Ce TP m’a permis d’acquérir une meilleure compréhension des bases de l’algorithmique à travers la manipulation de blocs visuels. J'ai utilisé une logique optimale pour maximiser l'efficacité des boucles et des conditions, ainsi qu’à m’adapter aux contraintes imposées.  
  
Ces notions constituent une étape pour progresser vers l’écriture d’algorithmes plus difficile s en langage de programmation.

