# **Rapport de Projet – Opération Flocon**

## **1. Présentation de l’équipe**

L’équipe de développement est composée de DUFOUR Raphael, MONTEIL Maxime et LEHEUDRE-EPSTEIN Lucien, étudiants en ingénierie à CY Tech. Ce projet a été réalisé dans le cadre de notre cours d’informatique (langage C), avec comme objectif la conception d’un jeu vidéo de type Tower Defense.

## **2. Sujet du projet et Description**

**Opération Flocon** est un jeu de tower defense se déroulant dans un univers hivernal. Le joueur doit défendre une couronne centrale contre des vagues d’ennemis composées de skieurs frénétiques, snowboarders acrobates et lugistes barjos. Pour se défendre, il peut compter sur des pingu-patrouilleurs, des flocons perce-ciel et des garde polaires qu’il peut positionner stratégiquement sur la carte. La carte de jeu est générée aléatoirement à chaque partie, avec un chemin unique menant à la couronne, que les ennemis empruntent automatiquement. Le joueur peut placer ses défenseurs autour de celui-ci, en tenant compte des contraintes de positionnement (les défenseurs peuvent être posés uniquement sur des cases de neige). L’objectif principal est de repousser les vagues successives d’ennemis en optimisant la position et le type des défenseurs, tout en gérant les ressources disponibles. Le joueur commence avec 350❄️ pour positionner ses défenseurs et avec 0 points, À chaque ennemi éliminé, il gagne un point et des flocons (❄️).

### **⚔️ Attaquants**

| **Nom** | **PV** | **Esquive** |
| --- | --- | --- |
| 🎿 **Skieur Frénétique** | 250 | 15% |
| 🏂 **Snowboarder Acrobate** | 500 | 30% |
| 🛷 **Lugiste Barjo** | 2000 | 01% |

### **🛡️Défenseurs**

| **Nom** | **Dégâts** | **Portée** |
| --- | --- | --- |
| 🐧**Pingu-Patrouilleur** | 50 | 5 |
| ⛄ **Flocon Perce-Ciel** | 300 | 10 |
| 🐻‍❄️ **Garde Polaire** | 100 | 1 |

## **3. Organisation de l’équipe et méthode de travail**

Notre équipe a adopté une méthode de travail efficace et coopérative. Nous avons commencé par choisir un thème à notre Tower Défense puis

* **Répartition des tâches** selon les forces et faiblesses des membres. Tout d’abord, Lucien et Raphaël se sont occupés de la carte et des fonctions permettant son affichage et sa création comme “creer\_chemin” ou “afficher\_carte” puisqu’ils sont à l’aise avec les tableaux et leur utilisation. Ensuite Maxime s’est occupé de la sauvegarde, du chargement ainsi que les fonctions étant proches de l’utilisateur comme la fonction “menu\_demarrage” par exemple puisqu’il est à l’aise avec les affichages et qu’il a bien compris les derniers chapitres du cours sur les fichiers.

**Les outils utilisés incluent :**

* Github comme dépôt de code accessible par les 3 membres du groupe
* Le langage C pour la programmation
* VirtualBox avec Debian pour travailler sur les ordinateurs personnels des développeurs.
* OBS Studio et Adobe Premiere Pro 2020 pour la vidéo de présentation.
* Discord pour les appels vocaux du groupe.
* Whatsapp comme groupe de discussion.

## **4. Problèmes rencontrés et solutions**

#### **Affichage en UTF-8 avec émojis**

### **Problème** : Certains caractères ne s’affichaient pas correctement selon les consoles (notamment sous Windows), ce qui a posé des problèmes, notamment lorsque l’on voulait travailler chez nous.

### **Solution** : Nous avons testé l’affichage sur plusieurs terminaux et vérifié que tous les émojis étaient bien encodés en UTF-8. Nous avons aussi ajusté les espacements pour préserver l’alignement visuel de la carte sur Linux. Actuellement, certains bugs d’affichage restent sur Windows ou sur d’autres consoles mais nous ne déplorons plus aucun bugs sur Linux. Pour pouvoir travailler chez nous, nous avons installé VirtualBox, nous permettant de simuler Linux sur un ordinateur Windows. Nous avons ainsi pu avancer sur le programme chez nous.

#### **Segmentation Fault persistant dans attaquer\_defenseur**

### **Problème** : Nous avons eu un souci dans la partie de la fonction “attaquer\_defenseur” qui utilise plusieurs tableaux précédemment initialisés dans le code et qui étaient mal initialisés.

### **Solution** : Nous avons retracé le parcours de ces tableaux dans le programme pour ensuite vérifier leur contenu avec des “printf” ce qui nous a permis de pouvoir ensuite régler le problème rencontré dans la fonction “attaquer\_defenseur” qui les utilisent.

#### **Fonctions fournies incompatible avec le programme**

### **Problème** : A chaque nouvelle fonction fournie par l’un des membres du groupe, le programme ne compilait plus, en effet, les membres donnaient des codes qui ne fonctionnaient pas bien avec les précédents, ou qui n'utilisaient pas les fonctionnalités dans le code précédent.

### **Solution** : Nous n’avons jamais vraiment trouvé une solution à ce problème, la seule chose que nous faisions était de reprendre les codes et de les modifier pour qu’ils compilent et fonctionnent avec le reste, quitte à le refaire du début. Cela nous a fait perdre énormément de temps malheureusement.

#### **Makefile et fichiers séparés (.h, .c, main …)**

### **Problème** : Nous avons eu du mal à faire le fichier Makefile car notre programme ne compilait pas, ce qui rendait impossible la vérification du Makefile. En plus, c'était la première fois que nous utilisions cela car nous l’avions seulement vu en cours.

### **Solution** : Nous avons demandé de l’aide à des personnes extérieures au groupe (Autres groupes, PRÉ ING2 et ING) qui ont pu nous expliquer le problème, nous aider à corriger le code ainsi que le solidifier.

#### **Fonction placement\_de\_defenseur**

### **Problème** : La fonction placement de défenseur a été un véritable calvaire à relier au reste du programme. En effet, cette fonction était pleine d’erreurs logiques, qui nous ont fait perdre un temps fou à corriger. La fonction compilait et marchait mais ne faisait pas du tout ce que nous voulions, elle faisait bugger les fonctions qui se servaient d’elle, ce qui nous a fait chercher des erreurs dans des fonctions ou il n’y en avait pas. La fonction a pris plus de 20 heures de codes à corriger.

### **Solution** : Maxime et Lucien ont passé le week-end du 10/05 et 11/05 à coder et tester la fonction en partie vocale sur Discord, ils y ont passé à peu près 16 heures. Grâce à l’aide d’un camarade de classe, ils ont pu corriger et faire fonctionner cette fonction avant la date limite. Permettant à l'équipe de faire fonctionner le code avant la date limite.

## **5. Commande de compilation avec gcc et exécution du programme**

## git clone <https://github.com/Snowsurf-007/Operation-Flocon>

## cd Operation-Flocon

## make

## **6. Remerciements**

L’équipe de développement tient à remercier Alexandre BERNARD (PRE ING 1 MI4 Groupe I) pour son aide sur le Makefile et l’aide fourni pour certaines corrections de bugs. Elle tient aussi à remercier Lucas BEZAMAT (ING1 GI) et Sergueï MANCHEC (PRE ING2 MI4) pour avoir regarder le programme et aider l’équipe à avoir un programme robuste.