Description des bonus:

après avoir eu quelques idées, j'ai décidé d'ajouter plusieurs bonus pour améliorer l'expérience de l'utilisateur.
Ci-dessous la liste et les détails de l'implementation des bonus ajoutés.

La liste des bonus:

- **La possibilité de soigner toute l'équipe du joueur**:
proposer d'effectuer un soin sur un personnage de son équipe ou de **soigner toute l'équipe en divisant les points de soin par le nombre de personnages vivants dans l'équipe**, y compris pour le mage qui va effectuer le soin.
J'ai choisi ce bonus pour qu'un personnage de type mage puisse rendre le jeu plus stratégique et ainsi pouvoir renverser le cours d'une partie en soignant toute son équipe plutôt qu'un seul personnage.

- **Ajout de 2 armes bonus et 2 équipements de soin bonus (les 4 se trouve dans le coffre) qui infligent des dégâts et des soins aléatoires**:
Les 2 armes supplémentaires bonus ajoutées sont: (le **drône** et le **C4**) qui infligent des dégâts aléatoires.
Les 2 équipements de soins bonus ajoutés sont: (**les petites potions** et l**es grandes potions**) qui régénèrent de manière aléatoire la vie du personnage à soigner s'il choisit de se soigner ou de soigner un membre de son équipe.
J'ai choisi ce bonus pour ajouter un peu plus de hasard au jeu.
Idem que pour le premier bonus, il permet de pouvoir renverser le cours d'une partie.

- **Affichage des statistiques en fin de partie:**
pour le dernier bonus, j'ai choisi d'**afficher les statistiques en fin de partie** (détails des actions de chaque personnage des deux équipes, coffre apparu…) car je trouve que les statistiques sont vraiment importantes, que ce soit dans un jeu ou une application car ça résume tout ce qu'il a pu se passer durant une partie dans un jeu ou l'utilisation d'un programme.
Cela encourage aussi l'utilisateur à vouloir s'améliorer ou à jouer plus stratégiquement à la prochaine partie et l'amène ainsi à rejouer.

L'implémentation des bonus:

- **La possibilité de soigner toute l'équipe du joueur**:

j'ai déclaré une méthode 'healMultiple()' qui prend pour paramètres (le joueur à qui appartient l'équipe a soigné et le personnage qui servira de soigneur pour soigner ses coéquipiers) à l'intérieur de la méthode jai utilisé une boucle for qui parcourt le tableau contenant les personnages du joueur et leur attribut à chaque tour de

boucle le nombre de pv que rend le soigneur divisé par le nombre de personnage en vie de l'équipe.

```
209     //method that is used to heal a chosen character
210     func healMultiple(playerAttacker: Players, healer: Characters) {
211         for character in playerAttacker.characters {
212             character.life += healer.healer! / playerAttacker.characters.count
213             communication.healInformation(player: playerAttacker, healer: healer, characterToHeal: character, healMultiple:
                    true)
214         }
215         print(communication.textSeparation)
216         playerAttacker.numberHealMultiple[healer.idNumber] += 1
217         playerAttacker.totalHealPV[healer.idNumber] += healer.healer!
218     }
```

- Ajout des 2 armes et 2 équipements bonus qui infligent des dégâts et des soins aléatoires:

Pour ce bonus je me suis servi de la méthode 'arc4random(retour d'une valeur aléatoire allant de 0 au nombre définit) + (nombre qui sera ajouté au nombre choisi par la méthode pour ainsi créer un interval personnalisé) '
J'ai fait cela pour que les armes bonus et les équipements bonus retournent une valeur aléatoire entre x et y lors d'une attaque ou d'un soin.

```
12      //dictionnary that contains bonus weapons and name weapons for all character types except the magician
13      let weaponsBonus: [WeaponsBonus: Int] = [.woodenArch: 14, .gun: 16, .uAV: Int(arc4random_uniform(3))+17, .c4:
            Int(arc4random_uniform(3))+19]
14      //dictionnary that contains bonus heal and names weapons for magician
15      let magicianBonus: [MagicianBonus: Int] = [.plasters: 14, .medicalKit: 16, .smallPotion: Int(arc4random_uniform(3))
            +17, .BigPotion: Int(arc4random_uniform(3))+19]
```

- **Affichage des statistiques en fin de partie:**

Pour afficher les statistiques en fin de partie j'ai déclaré une fonction que j'ai nommée displayStats() qui prend en paramètres le joueur à qui on veut afficher les statistiques, ce paramètre sert aussi à accéder au nom des perso de l'équipe). Á l'intérieur de cette fonction j'ai utilisé une boucle for qui affiche tous les statistiques d'un perso puis passe au suivant pour le second tour de boucle jusqu'à l'affichage des statistiques de toute l'équipe

```
    //method that displays the statistics at the end of the game
    func DiplaysStats(player: Players) {
        //index that displays the corresponding stats of each character at the end of the game
        var index = 0
        print(communication.sceneries1)
        print(communication.sceneries1)
        print("\nJoueur \(player.playerTeamNumber), \(communication.yourStats)\n")
        print("Nombre de coffre: \(player.totalNumberChest)\n")

        for _ in 0...2 {
            print(communication.sceneries4)
            print("\(player.charactersNames[index]):")
            print("Nombre D'attaques: \(player.numberAssault[index])")
            print("Total Dégâts: \(player.totalWeaponDamages[index])")
            if player.numberHeal[index] != 0 || player.numberHealMultiple[index] != 0 {
                print("Nombre de soins: \(player.numberHeal[index])")
                print("Nombre de soins multiples: \(player.numberHealMultiple[index])")
                print(communication.sceneries4 + "\n")
            } else {
                print(communication.sceneries4 + "\n")
            }
            index += 1
        }
        print(communication.sceneries1)
        print(communication.sceneries1 + "\n")
    }
```