SAÉ 1.01 - Implémentation d'un Besoin

1 Equipe

MAURICE Alexandre TD III – TP5

DARGAZANLI Nicolas TD III – TP5

2 Exercice traité : Pack 09

Jeu 1 : Deviner un nombre

3 Rappel des spécifications du programme

On demande d'afficher :

- Les règles du jeu
- Un message d'erreur si le joueur entre un nombre de saisies incorrect
- Les bornes de l'intervalle (qui devront être saisies par le joueur) et le nombre de tentatives autorisées
- Le numéro de la tentative en cours
- Un message de félicitation en cas de succès
- Un message d'échec en cas d'échec

Les extensions sont soulignées.

3.1 Spécifications initiales

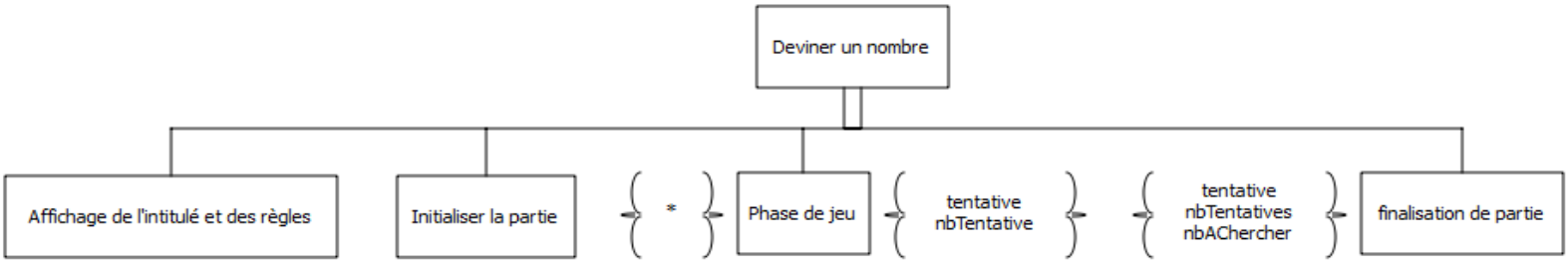
Ce programme devra, après avoir affiché les règles, demander l'intervalle dans lequel le joueur doit chercher un nombre, ainsi que le nombre de tentatives maximal que s'octroie le joueur. Ce dernier devra ensuite chercher un nombre dans cet intervalle grâce à des indications données par le programme (« C'est plus ! » ou « C'est moins ! ». Une fois que le joueur a trouvé le nombre ou qu'il a dépassé son nombre de tentatives, on lui annonce l'issue de la partie (victoire ou défaite), ainsi son nombre de tentatives utilisées s'il a gagné ou le nombre à chercher s'il a perdu.

3.2 Spécifications complémentaires = extensions traitées

Nous avons traité les deux extensions proposées. (Vérification de la saisie et saisie des bornes)

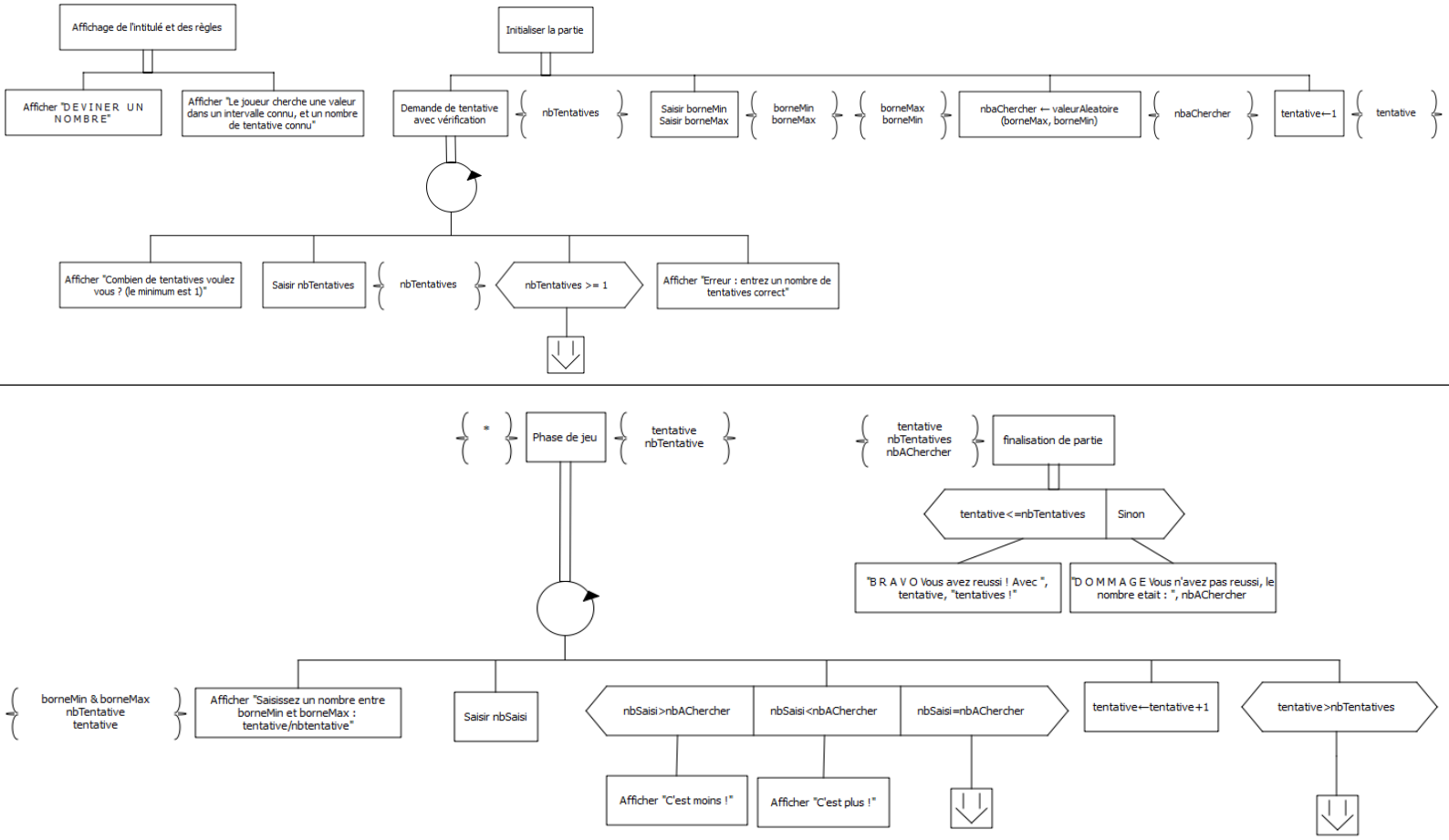
4 Algorithmes du programme (action principale et ses sous-actions)

Algorithme principal :



Ici, l'astérisque (*) correspond à des données pour le sous-problème « Phase de jeu » qui sont : borneMax ; BorneMin ; nbTentative ; nbAChercher ; tentative

Cet algorithme n'est qu'une division en sous-problèmes, et ces derniers doivent être détaillés à part, pour des causes de lisibilité lors de l'impression. Le sous-problème « Affichage de l'intitulé des règles » correspond alors à l'algorithme nommé de la même manière ci-dessous. De même pour les autres sous-problèmes.



Ici, l'astérisque (*) correspond à des données pour le sous-problème « Phase de jeu » qui sont : borneMax ; BorneMin ; nbTentative ; nbAChercher ; tentative

4.1 Affichage de l'intitulé et des règles

4.1.1 But de l'action

On affiche le nom du jeu et les règles afin que le joueur puisse comprendre ce qu'il doit faire.

4.1.2 Stratégie de l'algorithme mise en œuvre

Cela consiste en une décomposition séquentielle qui affiche d'abord le nom du jeu puis explique les règles.

4.2 Initialisation de la partie

4.2.1 But de l'action

Il s'agit d'entrer le nombre de tentatives (ainsi que de vérifier si ce dernier est correct), les bornes minimales et maximales, d'attribuer au nombre à chercher (nbaChercher) une valeur aléatoire puis enfin d'initialiser la variable tentative à 1. On justifie cela par rapport à l'incrémentation qui va suivre en 4.3.1.

4.2.2 Stratégie de l'algorithme mis en œuvre

Cet algorithme est conforme à une décomposition itérative non bornée suivie d'une décomposition séquentielle. La décomposition itérative consiste à une boucle qui s'effectue tant que la condition (nbTentative>=1) n'est pas vérifiée.

Ensuite, la décomposition séquentielle permet de saisir les bornes, puis en fonctions de ces bornes, de trouver un nombre aléatoire, et enfin d'initialiser la variable tentative à 1 (pour la première tentative du joueur).

4.2.3 Dictionnaire des éléments associés à cet algorithme

Nom	Type	Signification
nbTentative	Entier	Nombre de tentative maximal autorisé pour le joueur
borneMin	Entier	Borne minimale dans laquelle le joueur doit chercher
borneMax	Entier	Borne maximale dans laquelle le joueur doit chercher
nbAChercher	Entier	Nombre que le joueur doit chercher
tentative	Entier	Numéro de la dernière tentative actuelle du joueur

4.3 Phase de jeu

4.3.1 But de l'action

On rentre ici dans la « Phase de jeu » de l'algorithme principal. Ici, le joueur doit entrer un nombre compris dans l'intervalle puis essayer de deviner le nombre en se rapprochant de ce dernier. On affiche « C'est plus ! » si le nombre saisi est trop petit, et « C'est moins ! » si le nombre saisi est trop grand. On incrémente automatiquement le nombre de tentatives et l'on sort de la boucle non bornée quand le joueur a trouvé le nombre ou s'il a dépassé le nombre de tentatives autorisé.

4.3.2 Stratégie de l'algorithme mis en oeuvre

Cet algorithme est une décomposition itérative non bornée qui fera saisir au joueur un nombre, suivi de trois conditions mutuellement exclusives :

- Le nombre saisi est trop petit, auquel cas on affiche « C'est plus ! »
- Le nombre saisi est trop grand, auquel cas on affiche « C'est moins ! »
- Le nombre saisi est correct, auquel cas on sort de la boucle.

Puis à chaque tentative, on incrémente de 1 la variable tentative.

Si le joueur dépasse le nombre de tentatives autorisé, on sort de la boucle.
Cet algorithme peut donc se terminer de deux manières :

- Si on dépasse le nombre de tentatives
- Si on trouve le nombre cherché

4.3.3 Dictionnaire des éléments associés à cet algorithme

Nom	Type	Signification
nbTentative	Entier	Nombre de tentative maximal autorisé pour le joueur
borneMin	Entier	Borne minimale dans laquelle le joueur doit chercher
borneMax	Entier	Borne maximale dans laquelle le joueur doit chercher
nbAChercher	Entier	Nombre que le joueur doit chercher
tentative	Entier	Numéro de la dernière tentative actuelle du joueur
nbSaisi	Entier	Nombre saisi par le joueur

4.4 Finalisation de la partie

4.4.1 But de l'action

On doit ici finir la partie, c'est-à-dire dire si le joueur a gagné, auquel cas on doit spécifier son nombre de tentatives pour parvenir à la victoire, ou lui annoncer qu'il a perdu ainsi que le nombre qu'il était censé trouver.

4.4.2 Stratégie de l'algorithme mis en oeuvre

Cet algorithme suit une décomposition alternative de type si, sinon.

Si le nombre de tentatives est inférieur ou égal au nombre de tentatives maximal autorisé, le joueur gagne la partie, et le programme doit lui indiquer le nombre de tentatives utilisées. Sinon, le programme lui annonce qu'il a perdu, ainsi que le nombre qu'il était censé trouver.

Dans les deux cas, le programme s'arrête, correctement.

4.4.3 Dictionnaire des éléments associés à cet algorithme

Nom	Type	Signification
tentative	Entier	Numéro de la dernière tentative actuelle du joueur
nbAChercher	Entier	Nombre que le joueur doit chercher
nbTentative	Entier	Nombre de tentative maximal autorisé pour le joueur

5 Traces d'exécution

Scénario 1 : Entrée correcte pour le nombre de tentatives + le joueur gagne.

```
D E V I N E R   U N   N O M B R E
Le joueur cherche une valeur dans un intervalle connu, et en un nombre de tentative connu.

Combien de tentatives voulez-vous (le minimum est 1) ? 10

Entrez la borne minimale: 0
Entrez la borne maximale: 100
Saisissez un nombre entre 0 et 100. Tentative: 1/10 : 50
C'est moins !

Saisissez un nombre entre 0 et 100. Tentative: 2/10 : 25
C'est plus !

Saisissez un nombre entre 0 et 100. Tentative: 3/10 : 35
C'est moins !

Saisissez un nombre entre 0 et 100. Tentative: 4/10 : 30
B R A V O Vous avez reussi ! Avec 4 tentatives !
```

Scénario 2 : Entrée erronée pour le nombre de tentatives + le joueur perd.

```
D E V I N E R   U N   N O M B R E
Le joueur cherche une valeur dans un intervalle connu, et en un nombre de tentative connu.

Combien de tentatives voulez-vous (le minimum est 1) ? -1
Erreur : entrez un nombre de tentatives correct !

Combien de tentatives voulez-vous (le minimum est 1) ? 3

Entrez la borne minimale: 0
Entrez la borne maximale: 100
Saisissez un nombre entre 0 et 100. Tentative: 1/3 : 50
C'est plus !

Saisissez un nombre entre 0 et 100. Tentative: 2/3 : 75
C'est plus !

Saisissez un nombre entre 0 et 100. Tentative: 3/3 : 90
C'est plus !

D O M M A G E Vous n'avez pas reussi, le nombre etait : 100
```

6 Remarques

Informations que les étudiants souhaitent communiquer aux enseignants au sujet de cette SAÉ :

- Nous avons rencontré des difficultés avec le module random à la suite d'une erreur la première version donnée sur GitHub, problème que nous avons résolu suite à la deuxième version.
- Nous avons aussi rencontré des difficultés vis-à-vis de l'environnement de développement sur VS Code qui se lançait à l'origine en Powershell.

7 Code C++

Le fichier main.cpp est bien livré dans l'archive, avec les modules de la bibliothèque game-tools.