PIM : Mini-projet 1

Auteur 1 (Exercice 1 & 3) : Matthis Bernardini
Auteur 2 (Exercice 2) : Elisa Ciocarlan

Raffinages exercice 1

Les raffinages

R0 : Faire deviner à l'utilisateur un nombre choisi par l'ordinateur

R1 : **Comment** "Faire deviner à l'utilisateur un nombre choisi par l'ordinateur" ?

Choisir un nombre aléatoire x x: **out**

{ (0 < x < 1000) -- x compris entre 1 et 999 }

Faire deviner x à l'utilisateur en comptant le nombre de tentatives

R2 : **Comment** "Choisir un nombre aléatoire x" ?

X: Entier

Tirer_Un_Nombre_Aleatoire(X) x: **in**

R2 : **Comment** "Faire deviner x à l'utilisateur en comptant le nombre de tentatives ?

```
nb_tenta <-- 0
Répéter
    nb_tenta <-- nb_tenta + 1
    Demander un nombre y à l'utilisateur                y: out
    Indiquer à l'utilisateur si y est plus grand, plus petit ou égal à x
    Quitter Quand y vaut x
Fin Répéter
```

R3 : **Comment** "Demander un nombre y à l'utilisateur" ?

```
-- Non robuste conformément à l'énoncé.
Écrire("Choisir un nombre entre 1 et 999 inclus.")
Lire(y)
```

R3 : **Comment** "Indiquer à l'utilisateur si y est plus grand, plus petit ou égal à x" ?

```
Si y > x Faire
    Écrire("Trop grand !")
Sinon Si y < x Faire
    Écrire("Trop petit !")
Sinon
    Écrire("Trouvé en " + nb_tenta + "essai(s)")
FinSi
```

Evaluation par l'autre étudiant

		Evaluation Etudiant (I/P/A/+)	**Justification / commentaire**	**Evaluation Enseignant (I/P/A/+)**

Forme (D-21)	Respect de la syntaxe Ri : Comment "... une action complexe ..." ? des actions combinées avec des structures de controle Rj : ...	+		
	Verbe à l'infinitif pour les actions complexes	+		
	Nom ou équivalent pour expressions complexes	+		
	Tous les Ri sont écrits contre la marge et espacés	+		
	Les flots de données sont définis	+		
	Une seule décision ou répétition par raffinage	+		
	Pas trop d'actions dans un raffinage (moins de 6)	+		
	Bonne présentation des structures de contrôle	+		
Fond (D21-D22)	Le vocabulaire est précis	+		
	Le raffinage d'une action décrit complètement cette action	+		
	Le raffinage d'une action ne décrit que cette action	+		
	Les flots de données sont cohérents	+		
	Pas de structure de contrôle déguisée	+		
	Qualité des actions complexes	+		

Remarques diverses

Raffinages exercices 2

Les raffinages

R0: Trouver un nombre choisi par l'utilisateur et arrêter la partie si l'utilisateur triche

R1: Comment "Trouver un nombre choisi par l'utilisateur et arrêter la partie si l'utilisateur triche"?

Demander à l'utilisateur de choisir un nombre — Rep, nbChoisi: **out**

Déterminer le nombre choisi par l'utilisateur et vérifier qu"il ne triche pas — Triche, trouve, min, max: **out,** p, cpt: **in out**

Mettre fin à la partie — triche: **in**

R2: Comment "Demander à l'utilisateur de choisir un nombre"?

nbChoisi<-false
Tant que (non nbChoisi) **faire**
    Demander si l'utilisateur a choisi un nombre entre 1 et 999 — Rep: **out**
    Traiter la réponse de l'utilisateur — nbChoisi: **out** (boolean)
Fin Tant Que

R2: Comment "Déterminer le nombre choisi par l'utilisateur et vérifier qu"il ne triche pas"?

Initialiser les variables — min, max, trouve, triche, cpt : **out**
Repete
    cpt <- cpt+1
    Proposer un nombre à mi-chemin entre min et max — p: **out,** cpt: **in**
    Rep2: Character

    Traiter la réponse de l'utilisateur à la proposition — p: **in**; rep2, min, max, trouve: **out**
    Vérifier que l'utilisateur ne triche pas — min, max: **in**, triche: **out**
Tant que (non trouve OU non triche)

R2: Comment "Mettre fin à la partie"?

Si triche **faire**
    Ecrire ("Vous trichez. J'arrête cette partie")
Sinon faire
    Ecrire ("J'ai trouvé " + p + "en" + cpt + "essai(s).")
Fin Si

R3: Comment "Demander si l'utilisateur a choisi un nombre entre 1 et 999"?

Rep: Character
Ecrire(“Avez-vous choisi un nombre entre 1 et 999? (o/n)”)
Lire (Rep)

R3: Comment “Traiter la réponse de l’utilisateur”?
Si Rep=’o’ ou Rep=’O’ **faire** -- si l’utilisateur ne dit pas oui alors il n’a pas choisi
nbChoisi <- true
Sinon
Ecrire(“J’attends…”)
Fin Si

R3: Comment “Initialiser les variables”?
min <- 1 --Integer
max <- 999 --Integer
trouve <- false --Boolean
triche <- false --Boolean
cpt<-0 --Integer

R3: Comment “Proposer un nombre à mi-chemin entre min et max”?
p <- (min+max)/2
Ecrire (“Proposition” + cpt +”: “)
Ecrire (p)
Ecrire (“Trop (g)rand, trop (p)etit ou (t)rouvé? “)

R3: Comment “Traiter la réponse de l’utilisateur à la proposition”?
Lire (rep2)
Selon rep2 **faire**
‘g’, ‘G’ -> (min<-p-1)
‘p’, ‘P’ -> (max<-p+1)
‘t’, ‘T’ -> (trouve<-true)
Autres -> **Ecrire** (“Je n’ai pas compris. Merci de répondre: g si ma proposition est trop grande, p si ma proposition est trop petite, t si j’ai trouvé le bon nombre”)
Fin Selon

R3: Comment “Vérifier que l’utilisateur ne triche pas”?
Si (max<=min) **faire**
triche <- true
Fin Si

Evaluation par l’autre étudiant

		Evaluation Etudiant (I/P/A/+)	**Justification / commentaire**	**Evaluation Enseignant (I/P/A/+)**
Forme (D-21)	Respect de la syntaxe Ri : Comment "... une action complexe ..." ? des actions combinées avec des structures de controle Rj : ...	+		
	Verbe à l'infinitif pour les actions complexes	+		
	Nom ou équivalent pour expressions complexes	+		
	Tous les Ri sont écrits contre la marge et espacés	+		
	Les flots de données sont définis	+		
	Une seule décision ou répétition par raffinage	+		
	Pas trop d'actions dans un raffinage (moins de 6)	+		
	Bonne présentation des structures de contrôle	+		
Fond (D21-D22)	Le vocabulaire est précis	+		
	Le raffinage d'une action décrit complètement cette action	+		
	Le raffinage d'une action ne décrit que cette action	+		
	Les flots de données sont cohérents	+		
	Pas de structure de contrôle déguisée	+		
	Qualité des actions complexes	+		

Remarques diverses

Raffinages exercices 3

Les raffinages

R0 : Écrire un programme qui permet à l'utilisateur de jouer au jeu du devin contre l'ordinateur.

R1 : **Comment** "Écrire un programme qui permet à l'utilisateur de jouer au jeu du devin contre l'ordinateur" ?

Choix : Integer
Quit : Boolean

Quit <- False
Répéter

Permettre à l'utilisateur de choisir s'il souhaite deviner, faire deviner ou quitter.
Choix : **out**
Quit : **out**

Lancer l'action voulue par l'utilisateur
Quitter Quand Quit = True
Fin Répéter

R2 : Comment "Permettre à l'utilisateur de choisir s'il souhaite deviner, faire deviner ou quitter" ?

Ecrire("1- L'ordinateur choisit un nombre et vous le devinez")
Ecrire("2- Vous choisissez un nombre et l'ordinateur le devine")
Ecrire("0- Quitter le programme")
Lire(Choix)

R2 : Comment "Lancer l'action voulue par l'utilisateur" ?

Selon Choix **Faire**

1 => Faire deviner un nombre choisi par l'ordinateur **(cf Exercice 1)**

2 => Trouver un nombre choisi par l'utilisateur et arrêter la partie si l'utilisateur triche **(cf Exercice 2)**

0 => Quit <- True

Fin Selon

Evaluation par l'autre étudiant

		Evaluation Etudiant (I/P/A/+)	**Justification / commentaire**	**Evaluation Enseignant (I/P/A/+)**
Forme (D-21)	Respect de la syntaxe Ri : Comment "... une action complexe ..." ? des actions combinées avec des structures de controle Rj : ...	+		
	Verbe à l'infinitif pour les actions complexes	+		
	Nom ou équivalent pour expressions complexes	+		
	Tous les Ri sont écrits contre la marge et espacés	+		
	Les flots de données sont définis	+		
	Une seule décision ou répétition par raffinage	+		
	Pas trop d'actions dans un raffinage (moins de 6)	+		
	Bonne présentation des structures de contrôle	+		
Fond (D21-D22)	Le vocabulaire est précis	+		
	Le raffinage d'une action décrit complètement cette action	+		
	Le raffinage d'une action ne décrit que cette action	+		
	Les flots de données sont cohérents	+		
	Pas de structure de contrôle déguisée	+		
	Qualité des actions complexes	+		

Remarques diverses

Bilan

TODO : Dire quel bilan vous tirez de ce mini-projet (pour l'équipe et individuellement). Cette partie n'est pas prise en compte dans la notation !

Ce mini-projet nous a permis de découvrir un nouveau langage et de programmer en appliquant la méthode des raffinages.

Annexe : Le code complet

TODO : Copier/coller ici le code qui est sous PIXAL (Ctrl-A puis Ctrl-C sous PIXAL, puis Ctrl-V ici suffit). Attention, le code doit quand même être sur PIXAL pour les deux membres de l'équipe !

Evaluation du code

		Consigne : Mettre O (oui) ou N (non) dans la colonne Etudiant suivant que la règle a été respectée ou non. Une justification peut être ajoutée dans la colonne "commentaire".	
Commentaire	**Etudiant (O/N)**	**Règle**	**Enseignant (O/N)**
	O	Le programme ne doit pas contenir d'erreurs de compilation.	
	O	Le programme doit compiler sans messages d'avertissement.	
	O	Le code doit être bien indenté.	
	O	Les règles de programmation du cours doivent être respectées : toujours un Sinon pour un Si, pas de sortie au milieu d'une répétition...	
	O	Pas de code redondant.	
	O	On doit utiliser les structures de contrôle adaptées (Si/Selon/TantQue/Répéter/Pour)	

	O	Utiliser des constantes nommées plutôt que des constantes littérales.	
	O	Les raffinages doivent être respectés dans le programme.	
	O	Les actions complexes doivent apparaître sous forme de commentaires placés AVANT les instructions correspondantes, avec la même indentation	
	O	Une ligne blanche doit séparer les principales actions complexes	
	O	Le rôle des variables doit être explicité à leur déclaration (commentaire).	