PIM : Mini-projet 1

Auteur 1 (Exercice 1 & 3) : Arthur Bongiovanni
Auteur 2 (Exercice 2) : Léa Houot

Raffinages exercice 1

Les raffinages

R0 : Faire deviner un nombre généré aléatoirement à l'utilisateur

R1 : Comment “faire deviner un nombre généré aléatoirement à l’utilisateur” ?

- Générer un nombre aléatoire
  nbr_alea : OUT integer
- Faire deviner le nombre à l'utilisateur
  nb_essaie : OUT integer
  nbr_alea : IN
- Afficher le résultat
  nbr_alea : IN ; nb-essaie : IN

R2 : Comment “Générer un nombre aléatoire” ?

- Get_Random_Number(nbr_alea);
- Écrire(“J’ai choisi un nombre compris entre 1 et 999”);

R2 : Comment “faire deviner le nombre à l'utilisateur” ?

- nb_essaie <- 0;
- Repeter
  nb_essaie <- nb_essaie +1;
  Demander à l’utilisateur de proposer un nombre — proposition: OUT integer
  Donner une indication à l’utilisateur
  proposition: IN / nbr_alea: IN
  Jusqu’à proposition = nbr_alea;

R2 : Comment “afficher le résultat” ?

- Écrire(“Bravo. Vous avez trouvé” ,nbr_alea, “ en “, nb_essaie, “ essaie(s).”);

R3 : Comment “demander à l’utilisateur de proposer un nombre”?

- Écrire(“Proposition “, nb_essaie, “ : “);
- Lire(proposition);

R3 : Comment “donner une indication à l’utilisateur concernant sa proposition” ?

- Si proposition = nbr_alea Alors
  Écrire(“Trouvé.);
  Sinon si proposition < nbr_alea Alors
  Écrire(“Trop petit.”);
  Sinon
  Écrire(“Trop grand.”);
  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	A		
	Pas trop d'actions dans un raffinage (moins de 6)	A		
	Bonne présentation des structures de contrôle	A		
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

TODO : Indiquer ici ce qui est utile à l’enseignant pour comprendre les raffinages et/ou le programme correspondant à l’exercice 1. Cette partie peut être vide.

Raffinages exercices 2

Les raffinages

R0 : Deviner un nombre entre 1 et 999 fournis par l'utilisateur

R1 : Comment "deviner un nombre entre 1 et 999 fournis par l'utilisateur" ?

```
    Initialiser la borne minimale et la borne maximale du nombre à trouver
                                                        min, max, : OUT integer
    Initialiser le nombre d'essais                      nb_essais : OUT integer
    Demander demander à l'utilisateur pour commencer le jeu
    Répéter
        Deviner le nombre grâce à une recherche par dichotomie
                                                        nb_essais : IN OUT
                                                        Nombre : OUT integer
                                                        Trouvé : OUT boolean
                                                        Triche : OUT boolean
    Jusqu'à Trouvé ouAlors Triche
    Ecrire le résultat du jeu                           nombre, nb_essais : OUT
                                                        Triche, Trouve : IN
```

R2 : Comment "initialiser la borne minimale et la borne maximale du nombre à trouver" ?

```
    Min <- 1
    Max <- 999
```

R1 : Comment "initialiser le nombre d'essais" ?

```
    Nb_essais <- 0
```

R2 : Comment "demander à l'utilisateur pour commencer le jeu" ?

```
    Répéter
        Ecrire ("Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ?")
        Lire (Réponse)
        Si non Réponse ="o" et non Réponse="O" alors
            Ecrire ("J'attends…")
        Sinon
            Rien
        Fin Si
    Jusqu'à Réponse ="o" ou Réponse="O"
```

R2 : Comment "deviner le nombre" ?

```
    Proposer un nombre par dichotomie       nb_essais : IN OUT
```

Nombre : OUT

Répéter

Demander une indication à l'utilisateur — indication : OUT string

Traiter l'indication de l'utilisateur — indication : IN; Min, max : IN OUT; Trouvé : OUT; Indication_Comprise : OUT boolean

Jusqu'à Indication_Comprise

R3 : Comment "proposer un nombre par dichotomie" ?

Nb_essais <- nb_essais + 1
Nombre <- (max + min)/2
Ecrire ("Proposition", nb_essais, ":", nombre)

R3 : Comment "demander une indication à l'utilisateur" ?

Ecrire ("Trop (g)rand, trop (p)etit ou (t)rouvé ?")
Lire (Indication)

R3 : Comment "traiter l'indication de l'utilisateur" ?

Selon indication dans
"G" ou "g" alors
max <- nombre
"P" ou "p" alors
min <- nombre
"T" ou "t" alors
trouvé <- vrai
Autre alors écrire les indications que l'utilisateur peut donner
Fin Selon
Déterminer la compréhension de l'indication — Indication_comprise : OUT
Cherche une tricherie — Triche : OUT; Min, max, nombre : IN; Indication : IN

R4 : Comment "écrire les indications que l'utilisateur peut donner" ?

écrire ("Je n'ai pas compris. Merci de répondre :")
Écrire ("g si ma proposition est trop grande")
Écrire ("p si ma proposition est trop petite")
Écrire ("t si j'ai trouvé le nombre")

R4 : Comment "Déterminer la compréhension de l'indication" ?

Indication_comprise <- indication = "G" ou "g" ou "P" ou "p" ou "T" ou "t"

R4 : Comment "Chercher une tricherie" ?

Si nombre = min alors
    Triche <- indication = “G” ou “g”
SinonSi nombre = max alors
    Triche <- indication = “P” ou “p”
Sinon
    Rien
Fin si

R2 : Comment “écrire le résultat du jeu” ?
Si triche alors
    Ecrire (“Vous trichez. J’arrête cette partie.”)
Sinon
    Ecrire (“J’ai trouvé”, nombre, “en”, nb_essais, “essais”)
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 contrôle 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	A		
	Pas trop d'actions dans un raffinage (moins de 6)	P		
	Bonne présentation des structures de contrôle	A		
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

TODO : Indiquer ici ce qui est utile à l’enseignant pour comprendre les raffinages et/ou le programme correspondant à l’exercice 1. Cette partie peut être vide.

Raffinages exercices 3

Les raffinages

R0 : Jouer au jeu du devin avec l'utilisateur

R1 : Comment "jouer au jeu du devin avec l'utilisateur" ?
- programme_fini <- Faux; programme_fini : OUT bool
- Répeter
    - Demander le mode de jeu souhaité par l'utilisateur choix_jeu : OUT integer
    - Jouer au jeu choisi par l'utilisateur choix_jeu : IN

  Jusqu'à l'arrêt du programme programme_fini : IN
- Écrire("Au revoir…");

R2 : Comment "demander le mode de jeu souhaité par l'utilisateur" ?
- Répeter
    Présenter les modes de jeu possibles
    Demander à l'utilisateur de choisir un mode de jeu
    Déterminer la validité du choix choix_jeu : IN
    choix_est_valide : OUT bool

  Jusqu'à choix_jeu est correct choix_est_valide : IN

R2 : Comment "jouer au jeu choisi par l'utilisateur" ?
- Selon choix_jeu faire :
    - 1 => Faire deviner un nombre généré aléatoirement à l'utilisateur
    - 2 => Deviner un nombre entre 1 et 999 fournis par l'utilisateur
    - autre => programe_fini <- Vrai;

  Fin selon

R2 : Comment [déterminer] "l'arrêt du programme" ?
    Résultat <- programme_fini

R3 : Comment "Présenter les modes de jeu possibles" ?
- Écrire ("1 - L'ordinateur choisit un nombre et vous le devinez");
- Écrire ("2 - Vous choisissez un nombre et l'ordinateur le devine");

- Écrire (“0 - Quitter le programme”);

R3 : Comment “demander à l’utilisateur de choisir un mode de jeu” ?
- Écrire (“Votre choix ? : ”);
- Lire(choix_jeu);

R3 : Comment “Déterminer la validité du choix” ?
- Selon choix_jeu faire:
    - 0 .. 2 => choix_est_valide <- Vrai;
    - autre => choix_est_valide<- Faux;
      Écrire(“Choix incorrect”);

  Fin selon

R3 : Comment [déterminer] “choix_jeu est correct” ?
  Résultat <- choix_est_valide

R3 : Comment “faire deviner un nombre généré aléatoirement à l’utilisateur” ?
  Voire exercice 1

R3 : Comment “deviner un nombre entre 1 et 999 fournis par l’utilisateur” ?
  Voire exercice 2

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	A		
	Pas trop d'actions dans un raffinage (moins de 6)	A		
	Bonne présentation des structures de contrôle	A		
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

TODO : Indiquer ici ce qui est utile à l’enseignant pour comprendre les raffinages et/ou le programme correspondant à l’exercice 1. Cette partie peut être vide.

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 !

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.	
	N	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).	