PIM : Mini-projet 1

Auteur 1 Clément Demazure
Auteur 2 Adam Rochdi

TODO : Nommer votre document PIM-MP1-Équipe-XN où XN correspond au numéro d'équipe (voir "choisir mon équipe" sur Moodle).

Raffinages exercice 1

Les raffinages

TODO : écrire ici les raffinages pour l'exercice 1.

R0: "Faire deviner à nombre généré par ordinateur à un utilisateur"

R1: Comment :"Faire deviner à nombre généré par ordinateur à un utilisateur"

Initialiser la partie Nb_tentatives,Objectif :**in out** Entier Abandon, Reussite :**in out** Booléens

Répeter :

Faire une tentative de nombre Objectif :**in** Entier Nb_tentatives:**in out** Entier Abandon, Reussite :**out** Booléens

Jusqu'à ce que l'utilisateur trouve ou abandonne Abandon, Reussite :**in** Booléens

Afficher le résultat Nb_tentatives:**in** Entier Abandon, Reussite :**in** Booléens

R2: Comment :"Initialiser la partie"

Abandon **<-** Faux

Reussite **<-** Faux

Objectif **<-** Nombre aléatoire entre 1 et 999.

Nb_tentatives **<-** 0

R2: Comment :"Faire une tentative de nombre"

Demander un nombre à l'utilisateur Val_Entrée :**out** Entier Nb_tentatives:**in out** Entier

Comparer le résultat avec l'Objectif Val_Entrée :**in** Entier Reussite :**out** Booléens

R2: Comment [Déterminer] :"l'utilisateur trouve ou abandonne"

Résultat **<-** Reussite ou sinon Abandon

R2: Comment :"Afficher le résultat"

Si L'utilisateur a abandonné **Alors** Abandon :**in** Booléens

Écrire ("Vous avez abandonné, à une prochaine fois peut-être")

Sinon

Écrire ("Félicitation, vous avez gagné (en seulement ")

Écrire (Nb_tentatives)

Écrire ("Tentative.s)

FinSi

R3: Comment :"Demander un nombre à l'utilisateur"

Nb_tentatives **<-** Nb_tentatives+1

Écrire ("Choisissez un nombre")

Lire (Val_Entrée)

R3: Comment :"Comparer le résultat avec l'objectif"

Selon Val_Entrée **Faire**

1**..** Objectif-1	**=>** Demander un nombre plus grand	
Objectif+1**..**999	**=>** Demander un nombre plus petit	
0	**=>** Proposer à l'utilisateur d'abandonner	Abandon :**out** Booléens
Objectif	**=>** Reussite **<-**Vraie	

Autres => **Écrire** ("La valeur renseignée est incorrecte. Écrivez un nombre entre 1 et 999 où 0 si vous voulez abandonner.")

FinSelon

R4: Comment :"Demander un nombre plus grand"

Écrire ("Trop petit, réessayer.")

R4: Comment :"Demander un nombre plus petit"

Écrire ("Trop grand, réessayer.")

R4: Comment :"Proposer à l'utilisateur d'abandonner"

Écrire ("Voulez-vous abandonner? (o/*)")

Lire (Val_Entrée)

Si Val_Entrée = ‘o’

Abandon **<-** Vraie;

Sinon

Écrire ("Retour au jeu")

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 : ...	A		
	Verbe à l'infinitif pour les actions complexes	A		
	Nom ou équivalent pour expressions complexe il s	+		
	Tous les Ri sont écrits contre la marge et espacés	+		
	Les flots de données sont définis	A		
	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	P		
Fond (D21-D	Le vocabulaire est précis	A		

22)				
	Le raffinage d'une action décrit complètement cette action	A		
	Le raffinage d'une action ne décrit que cette action	P		
	Les flots de données sont cohérents	A		
	Pas de structure de contrôle déguisée	A		
	Qualité des actions complexes	A		

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

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

R1 : **Comment** "Faire deviner à l'ordinateur un nombre choisi par un utilisateur "
Commencer la partie en demandant un nombre
Répéter
L'ordinateur affiche un nombre
Jusqu'à ce que l'utilisateur confirme si c'est le nombre ou Détecter la triche
Afficher le résultat

R2: Comment "L'ordinateur affiche un nombre"
L'ordinateur utilisera la dichotomie pour trouver la valeur
Inf<—1 **–borne inf** Inf : **in** Entier
Sup<— 100 **–borne sup** Sup : **in** Entier
Mediane <—(sup-inf)div2 **–Valeur médiane** Mediane: **out** Entier

R3: **Comment** "l'utilisateur confirme si c'est le nombre"
Answer **: in** Charactère
Victoire : Booléan
Victoire <— False **—c'est une condition pour quitter la boucle**

Essais← 0 Essais : **out** Entier

–On utilisera la méthode de la dichotomie pour deviner le nombre

Répéter

Selon Answer **Dans**

“G” **ou** “g” => Sup← mediane **et** essais <— essais +1

“P” **ou** “p” => Inf← mediane **et** essais <— essais +1

Autres => Ecrire (“Saisissez un nombre valide“)

Jusqu’à = Victoire = true **ou** Essais > 10

R3: Comment “Savoir si l’utilisateur triche”

Si Essais > 10 **alors**

On quitte la boucle et

Ecrire (“ Vous trichez ! “)

–En utilisant la dichotomie en arrivera toujours à un nombre

–et on peut pas avancer encore, donc l’utilisateur .

R4: Comment (“Commencer la partie”)

Ecrire ("Le nombre " &(Mediane) & " est-il correct (C), trop petit (P), ou trop grand (G) ?");

R5: Comment (“Afficher le résultat “)

Ecrire(“J'ai trouvé ” + devine + ” en ” + essais+ ” essais.”)

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 : ...	A		
	Verbe à l'infinitif pour les actions complexes	A		
	Nom ou équivalent pour expressions complexes	A		
	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	+		
Fond (D21-D22)	Le vocabulaire est précis	A		
	Le raffinage d'une action décrit complètement cette action	P		
	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	A		

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

TODO : écrire ici les raffinages pour l'exercice 3.

R0: :"Jouer au jeu du devin."

R1: Comment :"Jouez au jeu du devin."
    Initialiser le jeu
    Répéter
        Afficher le menu principal

Choisir le mode de jeu Quitter :**out** Booléens
Terminer la partie Quitter :**out** Booléens
Jusqu'à ce que l'utilisateur quitte le jeu
Terminer le jeu

R2: Comment :"Initialiser le jeu"
Écrire ("Bienvenue sur le fameux jeu du devin")
Écrire ("Le but du jeu est d'essayer de deviner un nombre mystère compris entre 1 et 999, avec pour seuls indices les positions relatives des précédentes propositions")

R2: Comment :"Afficher le menu principal"
Écrire ("Sélectionnez votre mode de jeu")
Écrire ("Pour que l'ordinateur choisisse le nombre mystère taper 1")
Écrire ("Pour que l'ordinateur devine votre nombre mystère taper 2")
Écrire ("Pour quitter le jeu taper 0")

R2: Comment :"Choisir le mode de jeu"
Lire Val_Entrée
Selon Val_Entrée **Faire**
0 **=>** Quitter **<-** Vrai
1 **=>** Faire deviner à nombre généré par ordinateur à un utilisateur (cf exo 1)
2 **=>** Faire deviner à l'ordinateur un nombre choisi par un utilisateur (cf exo2)
Autres **=> Écrire** ("La valeur renseignée est incorrecte. Retour au menu principale.")
FinSelon

R2: Comment :"Terminer la partie"
Écrire ("Rejouer ? (o/*)")
Lire (Val_Entrée)
Si Val_Entrée = 'o'
Quitter **<-** Faux
Sinon
Quitter **<-** Vrai
FinSi

R2: Comment [Déterminer] :"l'utilisateur quitte le jeu"
Résultat **<-** Quitter

R2: Comment :"Terminer le jeu"
Écrire ("Merci d'avoir jouer, et à la prochaine")
Écrire ("Crédits :")
Écrire ("cet excellent jeu vous a été fourni par Adam Rochdi et Clément Demazure")

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 : ...	A		
	Verbe à l'infinitif pour les actions complexes	A		
	Nom ou équivalent pour expressions complexes	+		
	Tous les Ri sont écrits contre la marge et espacés	+		
	Les flots de données sont définis	A		
	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	P		
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	P		
	Les flots de données sont cohérents	A		
	Pas de structure de contrôle déguisée	A		
	Qualité des actions complexes	A		

Remarques diverses

Bilan

Ce projet nous a permis de mettre en pratique les cours sur les raffinages, de mieux comprendre la sémantique liée tout en se familiarisant un peu plus avec le langage Ada. De plus cet exemple concret nous a permis de comprendre l'intérêt du raffinage, à première vue redondant. Mais il permet au final d'éviter pas mal d'erreurs et d'anticiper une partie des difficultés tout en ayant un résultat (code) plus lisible grâce aux commentaires liés.

Annexe : Le code complet

```
with Text_Io;              use Text_Io;
with Ada.Integer_Text_Io;  use Ada.Integer_Text_Io;
with Alea;

-- Auteur : Clément Demazure
--

procedure Jeu_Devin_Exo1 is
   package Mon_Alea is
                  new Alea (2,999); -- générateur de nombre dans l'intervalle [1, 999]
   use Mon_Alea;
   Reussite: Boolean;-- Booléen indique si l'utilisateur a reussi à deviner
   Objectif: Integer;-- Entier indique le nombre que l'utilisateur doit deviner
   Nb_Tentatives: Integer;-- Entier pour compter le nombre de tentatives de l'utilisateur
   Val_Input: Integer;-- Entier pour stocker la dernière tentatives de l'utilisateur
begin -- Faire deviner à nombre généré par ordinateur à un utilisateur
     -- Initialiser la partie
   Put_Line("J'ai choisit un nombre, bonne chance pour le deviner!");
   Reussite := False;
   Get_Random_Number (Objectif);
   Nb_tentatives := 0;
   loop -- Faire une tentative de nombre
         -- Demander un nombre à l'utilisateur
         Put_Line("Choisissez un nombre");
         Get(Val_Input);
         -- Comparer le résultat avec l'objectif
         case Val_Input is
         when 1..999=> --l'utilisateur a bien essayé un nombre valide
                  Nb_tentatives := Nb_tentatives+1;
                  if Val_Input = Objectif then--l'utilisateur a gagné
```

```
                Reussite := True;
            elsif Val_Input < Objectif then-- Demander un nombre plus grand
                Put_Line("Trop petit, réessayer.");
            elsif Val_Input > Objectif then-- Demander un nombre plus petit
                Put_Line ("Trop grand, réessayer.");
            end if;
        when others => Put_Line("La valeur renseignée est incorrecte. Écrivez un nombre entre 1 et 999.");
        end case;
        exit when Reussite;
    end loop;
    -- Afficher le résultat
    Put("Félicitation, vous avez gagné (en seulement ");
    Put(Nb_tentatives);
    Put_Line("  Tentative.s");
end Jeu_Devin_Exo1;

—----------------------------------------------------------------------------------------------------------------------------------------fin exo1
with Text_Io;              use Text_Io;
with Ada.Integer_Text_Io;  use Ada.Integer_Text_Io;


-- Auteur : ROCHDI Adam
--
procedure Jeu_Devin_Exo2 is
        Inf, Sup, Mediane,Essais : Integer;
        Answer : Character;
        Victoire : Boolean;
begin
  Put_Line("Bienvenue au jeu de devinette, choisissez un nombre et laissez le reste sur moi !");
  Inf := 1;
  Sup := 1000; --Selon l'exercice le nombre est entre 1 et 999
  Victoire := False; --C'est pour quitter la boucle en cas de victoire
  Essais :=1; --Si ce nombre dépasse 9 alors l'utilisateur triche
  loop
        Mediane := (Inf + Sup) / 2; --J'ai procédé par la dichotomie
        Put_Line("Le nombre " & Integer'Image(Mediane) & " est-il correct (C), trop petit (P), ou trop grand (G) ?");
        Get(Answer); --Selon la réponse on va savoir quelle borne on doit changer
        case Answer is
        when 'C'|'c' => Victoire := True; --Je l'ai changer à True pour quitter la boucle
        when 'P'|'p' => Sup := Mediane ; Essais := Essais +1;
        when'G'|'g' => Inf := Mediane ; Essais := Essais +1;
        when others => Put_Line("Saisissez une lettre valide"); --Si l'utilisateur entre par erreur une autre lettre
        end case;
```

```
        exit when Victoire or Essais > 9;
        end loop;
        --On quitte la boucle soit à cause de la victoire ou la triche
        if Victoire then
        Put_Line("WOAH !J'ai trouvé le nombre " & Integer'Image(Mediane) & " en" & Integer'Image(Essais)& " Essais !
");
        else
        Put_Line ("Vous avez effectué beaucoup d'essais");
        end if;

end Jeu_Devin_Exo2;

——————————————————————————————————————————————————————————————————————————————————————————————————————————————fin exo2
with Text_Io;              use Text_Io;
with Ada.Integer_Text_Io;  use Ada.Integer_Text_Io;
with Jeu_Devin_Exo1;
with Jeu_Devin_Exo2;

-- Auteur : Clément Demazure
--

procedure Jeu_Devin_Exo3 is
   Val_Input: Integer;-- Entier pour stocker le choix dans le menu principale de l'utilisateur
   Val_Input_Chr: Character;-- Entier pour stocker le choix dans le menu pour rejouer de l'utilisateur
   Quitter: Boolean;
begin   --Jouez au jeu du devin
   --Initialiser le jeu
   Put_Line("Bienvenue sur le fameux jeu du devin");
   Put_Line("Le but du jeu est d'essayer de deviner un nombre mystère compris entre 1 et 999, avec pour seuls indices
les positions relatives des précédentes propositions");
   loop
         --Afficher le menu principal
         Put_Line("Sélectionnez votre mode de jeu");
         Put_Line("Pour que l'ordinateur choisisse le nombre mystère taper 1");
         Put_Line("Pour que l'ordinateur devine votre nombre mystère taper 2");
         Put_Line("Pour quitter le jeu taper 0");
         --Choisir le mode de jeu
         Get(Val_Input);
         case Val_Input is
         when 0 => Quitter := True;
         when 1 => Jeu_Devin_Exo1;
         when 2 => Jeu_Devin_Exo2;
         when others => Put_Line("La valeur renseignée est incorrecte. Voulez vous quitter le jeu ?");
```

```
            end case;
            --"Quitter le jeu ?"
            Put_Line("Continuer de jouer ? (o/*)");
            Get(Val_Input_Chr);
            if Val_Input_Chr = 'o' then
                  Quitter := False;
            else
                  Quitter := True ;
            end if;
            exit when Quitter;
      end loop;
--Terminer le jeu
Put_Line("Merci d'avoir jouer, et à la prochaine");
Put_Line("Crédits :");
Put_Line("cet excellent jeu vous a été fourni par Adam Rochdi et Clément Demazure");
end Jeu_Devin_Exo3;
```

—----------fin exo3

Evaluation du code

		Consigne : Mettre O (oui) ou N (non) dans la colonne Étudiant 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.	
	N	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	
	N	Une ligne blanche doit séparer les principales actions complexes	
	O	Le rôle des variables doit être explicité à leur déclaration (commentaire).	