PIM : Mini-projet 1

Auteur 1 (Exercice 1 & 3) : Baptiste RAJAONAH
Auteur 2 (Exercice 2) : Benjamin SOYER

TODO : Nommer votre document PIM-MP1-Equipe-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 l'humain un nombre choisi par l'ordinateur

Exemples:

```
1  J’ai choisi un nombre aléatoire entre 1 et 999.
2  Choisissez un nombre entre 1 et 999: 653
3  Le nombre est trop petit
4  Choisissez un nombre entre 1 et 999: 879
5  Le nombre est trop grand
6  Choisissez un nombre entre 1 et 999: 700
7  Le nombre est trop grand
8  Choisissez un nombre entre 1 et 999: 678
9  Vous avez trouvé
10 Nombre d’essais: 4
```

```
1  J’ai choisi un nombre aléatoire entre 1 et 999.
2  Choisissez un nombre entre 1 et 999: 500
3  Le nombre est trop petit
4  Choisissez un nombre entre 1 et 999: 798
5  Vous avez trouvé
6 Nombre d’essais: 2
```

R1: Comment "Faire deviner à l’humain un nombre choisi par l’ordinateur" ?

```
    Choisir un nombre aléatoire entre 1 et 999   nombre_aleatoire: out Entier
    Faire deviner le nombre à l'humain           nombre_essais: out Entier
    Afficher le nombre d'essais nécessaires      nombre_essais: in out Entier
```

R2: Comment "Choisir un nombre aléatoire entre 1 et 999" ?

```
    Choisir l'intervalle de valeur avec la bibliothèque Alea (Alea(min, max))
    Générer un nombre dans cet intervalle avec la fonction Get_Random_Number
    Ecrire(“J’ai choisi un nombre aléatoire entre 1 et 999.”)
```

R2: Comment "Faire deviner le nombre à l'humain" ?

```
    nombre_essais <- 0
    REPETER
        nombre_essais <- nombre_essais + 1
        Demander un nombre à l'humain   nombre_humain: out Entier
        {nombre_humain>=1 ET nombre_humain<=999}
        Indiquer si le nombre est trop grand, trop petit ou correct
    JUSQU'A nombre_aleatoire = nombre_humain
```

R2: Comment "Afficher le nombre d'essais nécessaires" ?

```
    Ecrire(“Nombre d’essais: “)
```

```
        Ecrire(nombre_essais)

R3: Comment "Demander un nombre à l'humain" ?
        Ecrire("Choisissez un nombre entre 1 et 999: ")
        Lire(nombre_humain)          nombre_humain: out Entier

R3: Comment "Indiquer si le nombre est trop grand, trop petit ou correct" ?
        SI nombre_humain>nombre_aleatoire ALORS
                Ecrire("Le nombre est trop grand")
        SI nombre_humain<nombre_aleatoire ALORS
                Ecrire("Le nombre est trop petit")
        SI nombre_humain = nombre_aleatoire ALORS
                Ecrire("Vous avez trouvé")
        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	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

TODO : écrire ici les raffinages pour l'exercice 2.
RO : Faire deviner à l'ordinateur le nombre de l'utilisateur.
Exemple:

Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ? o
Proposition 1 : 500
Trop (g)rand, trop (p)etit ou (t)rouvé ? g
Proposition 2 : 250
Trop (g)rand, trop (p)etit ou (t)rouvé ? x
Trop (g)rand, trop (p)etit ou (t)rouvé ? t
J'ai trouvé 250 en 2 essais.

Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ? x
J'attends...
Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ? O
J'attends...
44 Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ? O
Proposition 1 : 500
Trop (g)rand, trop (p)etit ou (t)rouvé ? T
J'ai trouvé 500 en 1 essai.

R1 : Comment faire deviner à l'ordinateur le nombre de l'utilisateur ?

Demander si l'utilisateur a choisi un nombre compris entre 1 et 999 jusqu'à que son choix soit effectif. confirmationChoix : out Caractère, choixNonConfirme : out Booléen

Initialiser l'intervalle des variables utilisées dans le fonctionnement du jeu
min, max : in out Entier
neTrichePAs, nombreNonTrouve, reponseUtilisateurNonValide : out Booléen
{max > min > 0 et neTrichePas, nombreNonTrouve, reoinseUtilisateurNonValide= Vrai}
Lancer le jeu dans l'objectif de trouver le nombre de l'utilisateur

R2 : Comment demander à l'utilisateur si il a choisi un nombre compris entre 1 et 999 ?

TANT QUE(choixNonConfirme) FAIRE
Ecrire("Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ?
Lire(confirmationChoix)
Actualiser la valeur de "choixNonConfirme" en fonction de la réponse utilisateur
fin TANT QUE

R2 : Comment initialiser les valeurs des variables utilisées dans le jeu ?

min <- 1
max <- 999
neTrichePas <- VRAI
nombreNonTrouve <- VRAI
reponseUtilisateurNonValide <- VRAI

R2 : Comment lancer le jeu dans l'objectif de trouver le nombre de l'utilisateur ?

TANT QUE neTrichePas et nombreNonTrouve FAIRE neTrichePas, nombreNonTrouve :inchoi
--L'algorithme s'arrête si l'ordinateur trouve le bon nombre ou si il détecte une triche

Proposer un nombre à l'utilisateur par la méthode dichotomique.
min, max : in Entier
propositionOrdinateur : out Entier
Actualiser le nombre d'essais de l'ordinateur dans le jeu nombreEssai : in out entier
Permettre à l'utilisateur de donner une réponse qui soit interprétable par le programme à la proposition de l'ordinateur.

nombreEssai : in Entier reponseUtilisateur : out Caractère

Vérifier que l'utilisateur ne triche pas.

min, max : in entier reponseUtilisateur : in caractère
neTrichePas : in out Booléen

Réagir à la réponse de l'utilisateur si il réussit ou non le jeu.

propositionOrdinateur : in entier
min, max: in out Entier
nombreNonTrouve : in out Booléen

fin TANT QUE

R3 : Comment actualiser la valeur de "choixNonConfirme" en fonction de la réponse utilisateur ?

```
SI choixConfirmation = "o"
    choixNonConfirme <- FALSE
    mi
Sinon
    Ecrire(“J’attends…)

Fin SI
```

R3 : Comment proposer un nombre à l'utilisateur par la méthode dichotomique ?

```
propositionOrdinateur <- (min+max)/2
Ecrire("Proposition ")
Ecrire(nombreEssai
Ecrire(propositionOrdinateur)
```

R3 : Comment actualiser le nombre d'essais de l'ordinateur dans le jeu ?

–R3 non nécessaire mais utilisé pour rendre le raffinage plus facilement lisible.

```
nombreEssai <- nombreEssai +1
```

R3 : Comment permettre à l'utilisateur de donner une réponse à la proposition de l'ordinateur ?

```
reponseUtilisateurNonValide <- VRAI –On réinitialise la variable car a chaque tour de boucle, on doit s’assurer que l’utilisateur a choisi un réponse compréhensible par le programme
TANT QUE(reponseUtilisateurNonValide)
    Ecrire("Trop (g)rand, trop (p)etit ou (t)rouve ?")
    Lire(reponseUtilisateur)
```

Vérifier que la réponse donné est compréhensible pour le programme
FIN TANT QUE

R3 : Comment vérifier que l'utilisateur ne triche pas ?

SI min=max et (reponseUtilisateur=g ou reponseUtilisateur=p) Alors
--On a ici un intervalle de recherche de taille 1 pour deviner le nombre de l'utilisateur, on a donc forcément la bonne réponse, donc on est certain que l'utilisateur ment si il déclare que l'ordinateur se trompe.

Ecrire("Vous trichez. J'arrête cette partie.")
neTrichePas <- FAUX
–Arret de la boucle du jeu à la découverte de la triche
SINON
Rien

Fin SI

R3 : Comment réagir à la réponse de l'utilisateur si il réussit ou non le jeu ?
–On suppose que l'utilisateur donne une réponse que l'ordinateur peut comprendre
SELON reponseUtilisateur DANS

"t" ou "T" =>
Ecrire("J'ai trouvé en ")
Ecrire(nombreEssai)
nombreNonTrouve <- FAUX

"g" ou "G" =>
max <- propositionOrdinateur+1

"p" ou "P" =>
min <- propositionOrdinateur-1

Autres => Rien

Fin SELON

R4 : Comment vérifier que la réponse donné est compréhensible par le programme ?

Selon reponseUtilisateur DANS

“T” ou “t” ou “p” ou “P” ou “g” ou “G” =>

reponseUtilisateurNonValide <- FAUX

Autres => RIen

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 3

Les raffinages

TODO :
R0: Jouer au jeu du devin avec l'ordinateur
1- L'ordinateur choisit un nombre et vous le devinez
2- Vous choisissez un nombre et l'ordinateur le devine
0- Quitter le programme
Votre choix : 1
J'ai choisi un nombre aléatoire entre 1 et 999.
Choisissez un nombre entre 1 et 999: 234
Le nombre est trop petit
Choisissez un nombre entre 1 et 999: 809
Le nombre est trop grand
Choisissez un nombre entre 1 et 999: 584
Le nombre est trop grand
Choisissez un nombre entre 1 et 999: 397
Vous avez trouvé
Nombre d'essais: 4

1- L'ordinateur choisit un nombre et vous le devinez
2- Vous choisissez un nombre et l'ordinateur le devine
0- Quitter le programme
Votre choix : 1
J’ai choisi un nombre aléatoire entre 1 et 999.
Choisissez un nombre entre 1 et 999: 667
Vous avez trouvé
Nombre d’essais: 1

1- L'ordinateur choisit un nombre et vous le devinez
2- Vous choisissez un nombre et l'ordinateur le devine
0- Quitter le programme
Votre choix : 2
Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ? o
Proposition 1 : 500
Trop (g)rand, trop (p)etit ou (t)rouvé ? g
Proposition 2 : 250
Trop (g)rand, trop (p)etit ou (t)rouvé ? t
J'ai trouvé 250 en 2 essais.

1- L'ordinateur choisit un nombre et vous le devinez
2- Vous choisissez un nombre et l'ordinateur le devine
0- Quitter le programme
Votre choix : 2

Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ? x
J'attends...
Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ? O
J'attends...
Avez-vous choisi un nombre compris entre 1 et 999 (o/n) ? O
Proposition 1 : 500
Trop (g)rand, trop (p)etit ou (t)rouvé ? T
J'ai trouvé 500 en 1 essai.

R1: Comment “Jouer au jeu du devin avec l’ordinateur” ?

    Initialiser la variable indiquant que le programme se répète en boucle
programmeOuvert : out booléen
    Tant que programmeOuvert Faire
        Faire jouer au mode de jeu choisi par l’humain
    FIN Tant que

R2 : Comment initialiser la variable indiquant que le programme se répète en boucle?

programmeOuvert <- VRAI

R2: Comment “Faire jouer au mode de jeu choisi par l’humain” ?

REPETER

Afficher les modes de jeu possibles

Demander le mode de jeu souhaité à l’utilisateur mode_jeu:out Entier

JUSQU’A mode_jeu = 1 OU mode_jeu = 2

Lancer le jeu choisi en fonction du choix de l'utilisateur mode_jeu:in Entier

R3: Comment “Afficher les modes de jeu possibles” ?

Ecrire(“0- Quitter le programme”)

Ecrire(“1- L'ordinateur choisit un nombre et vous le devinez”)

Ecrire(“2- Vous choisissez un nombre et l'ordinateur le devine”)

R3: Comment “Demander le mode de jeu souhaité à l’humain” ?

Ecrire(“A quel mode de jeu souhaitez-vous jouer ? ”

Lire(mode_jeu)

R3: Comment “Lancer le jeu choisi en fonction du choix de l’utillisateur” ?

SELON mode_jeu Dans

0 => programmeOuvert <- FAUX –Le programme se stop

1 => Faire deviner à l’utilisateur un nombre choisi par l’ordinateur

–Nous ne détaillons pas cette action car déjà raffiné dans l’exercice 1 . (voir exo1)

2 => Faire deviner à l’ordinateur un nombre choisi par l’utilisateur

–Nous ne détaillons pas cette action car déjà raffiné dans l’exercice 2. (voir exo2)

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.

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 !

Exercice 1:

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

-- Auteur : Baptiste RAJAONAH
--
-- TODO: à compléter...
procedure Jeu_Devin_Exo1 is

      function choix_nombre_aleatoire return Integer with
                  Post => choix_nombre_aleatoire'Result <= 999 and
choix_nombre_aleatoire'Result >= 1
                  --le nombre aleatoire est entre 1 et 999
      is
                  package Mon_Alea is new Alea(1, 999);
                  use Mon_Alea;
                  Nombre_Aleatoire: Integer;
      begin
                  Get_Random_Number(Nombre_Aleatoire);
                  return Nombre_Aleatoire;
      end choix_nombre_aleatoire;

      --procedure de test de la fonction choix_nombre_aleatoire
      procedure Tester_choix_nombre_aleatoire is

                  test_nombre_aleatoire: Integer;
      begin
                  test_nombre_aleatoire := choix_nombre_aleatoire;
                  pragma Assert(test_nombre_aleatoire <= 999 and
test_nombre_aleatoire >= 1);
      end Tester_choix_nombre_aleatoire;
```

```
	--fonction de comparaison entre nombre proposé et nombre aléatoire
	procedure comparer_nombre(nombre_ordi: in Integer; nombre_hum: in
Integer) is
			begin
				if nombre_hum > nombre_ordi then
					Put("Le nombre est trop grand.");
				elsif nombre_hum < nombre_ordi then
					Put("Le nombre est trop petit.");
				else
					Put("Bravo! Vous avez trouvé!");
				end if;
			end comparer_nombre;


	--declaration variables globales
	nombre_essais : Integer := 0;
	nombre_humain : Integer;
	nombre_aleatoire : Integer;

begin
	nombre_aleatoire := choix_nombre_aleatoire;
	loop
			nombre_essais := nombre_essais + 1;
			New_Line;
			Put("Choisissez un nombre entre 1 et 999: ");
			Get(nombre_humain);
			comparer_nombre(nombre_aleatoire, nombre_humain);
	exit when nombre_aleatoire = nombre_humain;
	end loop;

end Jeu_Devin_Exo1;
```

Exercice 2 :

```
1 with Text_IO; use Text_IO;
2 with ada.integer_text_io; use ada.integer_text_io;
3
4 procedure ex2_code is
5
6 neTrichePas : Boolean:=True;
```

```
 7 choixNonConfirme : Boolean:=True;
 8 nombreNonTrouve : Boolean:=True;
 9 reponseUtilisateurNonValide: Boolean:=True;
10
11 min : Integer :=1;
12 max : Integer :=999;
13 propositionOrdinateur : Integer;
14 nombreEssai : Integer :=0;
15
16
17 confirmationChoix : Character;
18 reponseUtilisateur : Character;
19
20 begin
21   --Demander si l'tuilisateur a choisi un nombre compris entre 1 et 999 jusqu'à que son choix
soit effectif
22   While(choixNonConfirme) loop
23
24      Put("Avez-vous choisi un nombre compris entre 1 et 999 (o/n)");
25      New_Line;
26      Get(confirmationChoix);
27      Skip_Line;
28
29      if confirmationChoix='o' then
30      choixNonConfirme:=False;
31      --L'utilisateur  confirme que son choix est effectif, on arrête donc la boucle
32      else
33      Put("J'attends");
34      New_Line;
35
36      end if;
37
38   end loop;
39
40   --L'initalisation des variables min, max, neTrichePas, nimbreNonTrouve a déjà eu lieu au
début du programme.
41
42   --LAncement du jeu
43   while(neTrichePas and nombreNonTrouve) loop
44      --recherche de solution par la méthode dichotomique
45      propositionOrdinateur:= (min+max)/2;
46      Put("Proposition ");
47      Put(nombreEssai);
```

```
48      Put(" ");
49      Put(propositionOrdinateur);
50      New_Line;
51
52      nombreEssai:=nombreEssai+1;
53      reponseUtilisateurNonValide:=True; --On réinitialise le booléen à chaque tour de boucle
pour avoir la confirmation d'une réponse valide pour chaqque tour de boucle
54
55      --Permet à l'tuilisateur de donner une réponse à la proposition de l'ordinateur, la boucle
s'arrête a la condition d'une réponse compréhensible par l'ordinateur
56      while(reponseUtilisateurNonValide) loop
57      Put("Trop (g)rnad, trop (p)etit ou (t)rouve ?");
58      New_line;
59      Get(reponseUtilisateur);
60      Skip_Line;
61
62      case reponseUtilisateur is
63
64      when 't'|'T'|'g'|'G'|'p'|'P' =>
65              reponseUtilisateurNonValide:=False;
66
67      when others =>
68              Null;
69
70      end case;
71
72      end loop;
73      --Vérifiaction que l'utilisateur ne triche pas
74      if(min=max) and (reponseUtilisateur='g' or reponseUtilisateur='p') then
75      --On a ici un intervalle de recherche de taille 1 pour deviner le nombre de l'utilisateur, on
a donc forcément la bonne réponse, donc on est certain que l'utilisateur ment si il déclare que
l'ordinateur se trompe
76
77      Put("Vous trichez. J'arrête cette partie.");
78      New_Line;
79      neTrichePas:=False;
80
81      else
82      null;
83
84      end if;
85
86      --réaction du programme à la réponse de l'utilisateur
```

```
87      case reponseUtilisateur is
88
89      when 't' | 'T' =>
90      --cas de la réussite du jeu
91      Put("J'ai trouvé en ");
92      Put(nombreEssai);
93              Put(" tentative !");
94              New_Line;
95      nombreNonTrouve:= False;
96
97      when 'g' | 'G' =>
98      --cas d'un nombre proposé trop grand
99              max:= propositionOrdinateur;
100
101     when 'p' | 'P' =>
102     --cas d'un nombre proposé trop petit
103             min:= propositionOrdinateur+1;
104
105     when others =>
106     null;
107
108     end case;
109
```

Exercice 3

```
with Text_Io;           use Text_Io;
with Ada.Integer_Text_Io;  use Ada.Integer_Text_Io;
with Alea;
with Jeu_Devin_Exo1;
with Jeu_Devin_Exo2;

-- Auteur : Baptiste RAJAONAH
procedure Jeu_Devin_Exo3 is
      --la procedure affiche les modes de jeu
      procedure afficher_modes_jeu is
      begin
                  Put("0 - Quitter le programme");
                  New_Line;
                  Put("1 - L'ordinateur choisit un nombre et vous le
devinez.");
                  New_Line;
```

```
                  Put("2 - Vous choisissez un nombre et l'ordinateur le
devine.");
                  New_Line;
      end afficher_modes_jeu;

      --declaration de variables globales
      programme_ouvert: Boolean := True;
      mode_jeu : Integer;
begin
      while programme_ouvert loop
                  loop
                        afficher_modes_jeu;
                        Put("A quel mode de jeu souhaitez-vous jouer ?");
                        Get(mode_jeu);
                  exit when mode_jeu = 0 or mode_jeu = 1 or mode_jeu = 2;
                  end loop;
                  case mode_jeu is
                        when 0 => programme_ouvert := False;
                        when 1 => Jeu_Devin_Exo1;
                        when 2 => Jeu_Devin_Exo2;
                        when others => null;
                  end case;
      end loop;
end Jeu_Devin_Exo3;
```

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).	