Eléments de correction

janvier 2016

1 - Génération aléatoire

[*barême indicatif : 5 pts*]

Question 1-1 :

```
1  -- fichier generation_aleatoire.adb
2
3  with ada.numerics.discrete_random;
4
5  package body generation_aleatoire is
6
7     package Alea is new ada.numerics.discrete_random(natural);
8     G : Alea.generator;
9
10    -- procédure qui initialise le générateur aléatoire avec le germe a
11    procedure init(a : integer) is
12    begin
13       Alea.reset(G, a);
14    end init;
15
16    -- fonction qui génère un entier (1, 2 ou 3)
17    -- suivant deux probabilités p1 et p2 exprimées en pourcentage
18    --    la valeur 1 est générée avec une probabilité égale à p1 %
19    --    la valeur 2 est générée avec une probabilité égale à p2 %
20    --    la valeur 3 est générée avec une probabilité égale à 100-p1-p2 %
21    -- Précondition : p1+p2<=100
22    function alea_1_2_3(p1, p2 : pourcentage) return natural is
23       res, v : natural;
24    begin
25       v := Alea.random(G) mod 100;
26       if v<p1 then
27          res := 1;
28       elsif v<p1+p2 then
29          res := 2;
30       else
31          res := 3;
32       end if;
33       return res;
34    end alea_1_2_3;
35 end generation_aleatoire;
```

Question 1-2 :

```
1        -- creer le fichier sequence sequence de caracteres
2        create(f, out_file, argument(4));
3        -- initialiser le générateur
4        init(germe);
5        -- générer la suite de caractères
6        for i in 1..n loop
7           case alea_1_2_3(p1,p2) is
8              when 1 => put(f, 'A');
9              when 2 => put(f, 'D');
10             when 3 => put(f, 'G');
11             when others => null;
12          end case;
13       end loop;
14       -- fermer le fichier
15       close(f) ;
```

2 - Exploitation de la séquence créée

[*barême indicatif : 5 pts*]

```
      -- lire fichier  de caracteres et construire les statistiques
      -- A COMPLETER (Question 2.1)
      initstats ;
      open(f, In_FIle, argument(1)) ;
      while not end_of_file(f) loop
         get(f, c) ;
         case c is
            when 'A' => if Av then v := v+1 ;
                        else Av := true ; v := 1 ;
                        end if ;
            when 'G' | 'D' => if Av then Av := false ;
                                         ajoutervaleur(v) ;
                              end if ;
            when others => null ;
         end case ;
      end loop ;
      -- récupérer la dernière séquence de A
      if Av then ajoutervaleur(v) ; end if ;
      Close(f) ;

      -- afficher les statistiques
      -- A COMPLETER (Question 2.2)
      put("LG :  NB") ; new_line ;
      for i in 1..10 loop
         put(integer'image(i)&" : "&integer'image(effectif(i))) ;
         new_line ;
      end loop ;
```

3 - Robots

[*barême indicatif : 5 pts*]

Question 3-1 :

Dans ***systeme_robot.ads*** :

```
   function suivant return character ;
```

Dans ***systeme_robot.adb*** :

```
        function suivant return character is

                ar : ActionRobot;
                cc : ConfigCase;
-- DEBUT MODIF QUESTION 3.1
                ch : character ;
-- FIN MODIF QUESTION 3.1
        begin
                cc := config_case_devant;
                faire_transition(a, cc, ar);
                case ar is

-- DEBUT MODIF QUESTION 3.1
                   when AVANCE => avancer; ch := 'A' ;
                   when DROITE => tourner_a_droite; ch := 'D' ;
                   when GAUCHE => tourner_a_gauche; ch := 'G' ;
-- FIN MODIF QUESTION 3.1

                end case;
                nb_t := nb_t+1;
-- DEBUT MODIF QUESTION 3.1
                return ch ;
-- FIN MODIF QUESTION 3.1
        end suivant;
```

Question 3-2 :

```
1  -- programme pour évaluer un robot automate dans une suite de terrains
2  -- syntaxe : test_performance fichier_automate fichier_terrains fichier_resultat
3  --  avec : fichier_automate = fichier contenant la description d'un automate
4  --         fichier_terrains = fichier contenant une suite de terrains
5  --         fichier_resultat = fichier dans lequel sont ecrits les statistiques
6  --          du robot. fichier texte avec le format suivant
7  --             nb_de_terrains
8  --             sur chaque ligne, le nombre de transitions utilisé par le robot
9  --               pour sortir du terrain ou -1 si le robot n'est pas sorti
10
11 with ada.text_io, ada.integer_text_io, ada.float_text_io, ada.command_line;
12 with systeme_robot;
13 use ada.text_io, ada.integer_text_io, ada.float_text_io, ada.command_line;
14
15 procedure test_performance is
16
17    procedure process is
18
19       type EtatRobot is (Marche,Bloque,Sorti);
20
21       fT : file_type; -- fichier des terrains
22       fR : file_type; -- fichier des résultats
23       nb_terrains : positive;
24       boucle : boolean := true;
25       er : EtatRobot;
26
27 -- DEBUT MODIF QUESTION 3.2
28       ch : character ; -- le résultat de la fct suivant
29       lg, lgmax : integer ; -- lg courante et lg max de seq de A
30       AV : boolean ; -- vrai pendant les séquences de A
31 -- FIN MODIF QUESTION 3.2
32
33    begin
34       -- ouverture du fichier des terrains
35       open(fT, in_file, argument(2));
36
37       -- ouverture du fichier des resultats
38       create(fR, out_file, argument(3));
39
40       -- lecture du nombre de terrains
41       get(fT, nb_terrains);
42       put(fR, nb_terrains); new_line(fR);
43
44       for i in 1..nb_automates loop
45
46 --  DEBUT MODIF QUESTION 3.2
47          Av := false ;
48          lgmax := 0 ;
49          lg := 0 ;
50 -- FIN MODIF QUESTION 3.2
51
52          -- initialiser le SystemeRobot avec l'automate et le terrain
53          systeme_robot.init(argument(1), fT) ;
54
55          -- le SystemeRobot est en état de marche
56          er := Marche;
57
58          -- boucle sur les transitions du robot
59          while er=Marche loop
60
61             -- effectuer une transition
62             --systeme_robot.suivant;
63
64
65
66
67
68
```

```
69 -- DEBUT MODIF QUESTION 3.2
70             ch := systeme_robot.suivant ;
71             case ch is
72                when 'A' => if Av then lg := lg +1 ;
73                            else
74                              AV := true ;
75                              lg := 1 ;
76                            end if ;
77                when 'G' | 'D' => if Av then
78                                    Av := false ;
79                                    if lg > lgmax then
80                                        lgmax := lg ;
81                                        lg := 0 ;
82                                     end if ;
83                                  end if ;
84                when others => null ;
85             end case ;
86 -- FIN MODIF QUESTION 3.2
87
88             -- verifier si le robot est sorti ou
89             --  si le nb max de transitions est atteint
90             if systeme_robot.est_sorti then
91                boucle := false;
92                er := Sorti;
93             elsif systeme_robot.max_transitions_atteint then
94                boucle := false;
95                er := Bloque;
96             end if;
97
98          end loop;
99
100 -- DEBUT MODIF QUESTION 3.2
101          if Av then
102             if lg > lgmax then lgmax := lg ; end if ;
103          end if ;
104 -- FIN MODIF QUESTION 3.2
105
106          -- ecrire le resultat
107          if er=Sorti then
108
109 -- DEBUT MODIF QUESTION 3.2
110             --put(fR, systeme_robot.nb_transitions,width=>1);
111             put(fr, integer'image(lgmax)) ;
112 -- FIN MODIF QUESTION 3.2
113
114          else
115             put(fR, "-1");
116          end if;
117          new_line(fR);
118
119       end loop;
120
121       -- fermeture des fichiers
122       close(fR);
123       close(fA);
124
125    end process;
126
127 begin
128    if argument_count /= 3 then
129       put("syntaxe : " & command_name);
130       put_line(" fichier_automate fichier_terrains fichier_resultat");
131    else
132       process;
133    end if;
134 end test_performance;
```

4 - Histogramme

[*barême indicatif : 5 pts*]

Question 4-1 :

– D'après la figure, il faut que :

$$\texttt{LF} = (3 \times \texttt{NL} - 1) \times \texttt{L\_RECT} + 2 \times \texttt{MARGE} \leq \texttt{L\_MAX}$$

$$\iff \texttt{L\_RECT} = \left\lfloor \frac{\texttt{L\_MAX} - 2 \times \texttt{MARGE}}{3 \times \texttt{NL} - 1} \right\rfloor$$

avec $\lfloor\ \rfloor$ la partie entière inférieure.
– Avec `NL=10`, `MARGE=10` et `L_MAX=600` on obtient

$$\texttt{L\_RECT} = \left\lfloor \frac{600 - 2 \times 10}{3 \times 10 - 1} \right\rfloor$$

Soit `L_Rect` $= 20$.

Question 4-2 :

En ligne 69 :

```
1   -- tracé des classes
2   if e_max>0 then
3      for i in 1..NL loop
4         -- rectangle robot A
5         XR_min := XC_min+(i-1)*(3*L_RECT);
6         XR_max := XR_min+L_RECT;
7         YR_min := Y_max+((Y_min-Y_max)*StatRobotA(i))/e_max;
8         YR_max := Y_max;
9         ChangerCouleur(jaune);
10        RectanglePlein(XR_min,YR_min,XR_max,YR_max);
11        ChangerCouleur(noir);
12        Rectangle(XR_min,YR_min,XR_max,YR_max);
13
14        -- rectangle robot B
15        XR_min := XC_min+(i-1)*(3*L_RECT)+L_RECT;
16        XR_max := XR_min+L_RECT-1;
17        YR_min := Y_max+((Y_min-Y_max)*StatRobotB(i))/e_max;
18        YR_max := Y_max-1;
19        ChangerCouleur(vert);
20        RectanglePlein(XR_min,YR_min,XR_max,YR_max);
21        ChangerCouleur(noir);
22        Rectangle(XR_min,YR_min,XR_max,YR_max);
23     end loop;
24  end if;
```