

Nom :

Prénom :

DS POO 2017-2018

2ème session

L'objectif du sujet est de réaliser un programme permettant à une agence de transport de gérer ses colis.

Il est indispensable de lire le sujet dans son ensemble avant de commencer à répondre.

Le sujet comporte plus de points que nécessaire pour obtenir la note maximale.

Des bonus sont attribués lorsque vous traitez correctement une partie complète.

Le code partiel de ce programme est donné en fin de sujet, il vous appartient de le compléter en fonction des questions. Les questions sont indépendantes mais il est préférable de les traiter dans l'ordre.

Les étoiles (*, **, ***) indiquent le niveau de difficulté de la question (facile, moyen, difficile).

1 Cahier des charges :

Une agence de transport dispose de moyens de transport permettant d'acheminer des colis.

Un moyen de transport est capable de transporter un ou plusieurs types de colis.

Ces colis ont différentes caractéristiques, qui affectent leur mode de transport, les délais de livraison et le coût.

Dans ce sujet, nous traiterons uniquement 3 caractéristiques possibles, mais il peut y en avoir d'autres :

- colis nécessitant un transport frigorifique ;
- colis nécessitant un transport urgent ;
- colis transportés en mode économique.

Un colis peut combiner plusieurs caractéristiques : frigorifique ET urgent, urgent ET économique, etc.

Par défaut, un colis est transporté en mode économique (pour simplifier).

Il est possible qu'une agence ne dispose pas de moyen de transport pour expédier un colis (dans la classe Test, l'agence n'a pas de transport frigorifique).

2 Analyse des solutions possibles (16 points)

Une première solution consisterait à faire une hiérarchie de Colis (ColisUrgent, ColisUrgentEtFroid, etc.).

2.1 Pourquoi cette solution n'est-elle pas souhaitable ? (2 pt) *

Une autre solution consisterait à utiliser des attributs booléens dans la classe Colis, indiquant si un mode de transport doit être utilisé :

```
1 public class Colis
2 {
3     private boolean froid;
4     private boolean urgent ;
5     private boolean eco;....
```

2.2 Pourquoi cette solution n'est-elle pas souhaitable non plus ? (2 pt) *

La solution retenue (cf.code donné) consiste à associer aux modes de transport et aux colis des comportements (ComportementColis).

2.3 Dessinez l'organisation des classes et des interfaces du code donné. Distinguez les abstractions (classes abstraites, interfaces) des classes concrètes (2 pt) *

2.4 Quelle est l'utilité de l'organisation en packages et sous-packages ? (1pt) *

2.5 A quoi sert ComportementConcret ? Pourquoi implémente-t-elle Comparable(3pt) **

2.6 Pourquoi utilise-t-on un Set comme attribut `comportementsPossibles` dans Agence ? (1pt) **

2.7 Pourquoi déclare-t-on des List et des Set et non pas des TreeSet et des ArrayList ? (2pt) **

2.8 Quel peut être l'intérêt que Colis implémente ComportementColis, alors que cette classe contient déjà un attribut listant les comportements possible pour le colis (3pt) ***

3 Codage du Modèle (26 points)

Les espaces laissés vides sont à compléter. Leur taille devrait suffire pour contenir votre code.

Parfois, seule une petite partie de cet espace est nécessaire.

Il faut respecter les import déclarés (pas d'autre import nécessaires).

Le modèle contient les classes qui modélisent le problème sans gérer la partie IHM.

LISEZ BIEN LES COMMENTAIRES QUI DONNENT DES INDICES SUR CE QUI DOIT ÊTRE CODÉS ! Certaines parties du code sont très faciles, d'autres beaucoup plus dures !

3.1 Classe Agence (8 points) /**

```
6 package ds4eti2018_2S.modele;
7
8 import java.util.HashMap;
9 import java.util.Iterator;
10 import java.util.List;
11 import java.util.ArrayList;
12 import java.util.Map;
13 import java.util.Set;
14 import java.util.TreeSet;
15
16 import ds4eti2018_2S.modele.comportements.ComportementColis;
17 import ds4eti2018_2S.modele.transport.MoyenTransport;
18
19 public class Agence {
20     private String nom;
21     private String ville;
22     private Set<ComportementColis> comportementsPossibles = new
23     TreeSet<ComportementColis>();
24     private List<MoyenTransport> transports;
25     private List<Colis> lesColis = new ArrayList<Colis>();
26
27     public Agence(String nom, String adresse, List<MoyenTransport> transports){
28         this.nom=nom;
29         this.ville=adresse;
30         this.transports=transports;
31         if(transports!=null){majComportements();}
32     }
33     //constructeur ne donnant pas de liste de moyen de transport
34     public Agence(String nom, String adresse){
35
36     }
37     private void majComportements() {
38         //ajouter les comportements des transport à la liste des comportements possibles
39         for(MoyenTransport m:transports){
40
41         }
42     }
43
44     public boolean addColis(Colis c){
45         boolean ok=true;
46         Iterator<ComportementColis> it = c.getCaracteristiques().iterator();
```

```

45     while(ok && it.hasNext()){
46 //vérifier que l'agence a bien les compétences pour prendre en charge le colis
47     }
48     if(ok){lesColis.add(c);}
49     return ok;
50 }
51
52 public void affecteChargement(){
53     for(Colis c:lesColis){
54         boolean trouve=false;
55         Iterator<MoyenTransport> it = transports.iterator();
56         while(!trouve && it.hasNext()){
57
58         }
59     }
60
61     public double calculPrix(int km){
62         double prix=0;
63         for(Colis c:lesColis){prix+=c.getPrix(km);}
64         return prix;
65     }
66
67     public String toString(){
68         String cmp="";
69         for(ComportementColis cc:comportementsPossibles){
70             cmp+="\t"+cc+"\n";
71         }
72         return "Agence "+nom+ " située à "+ville+"\n"+cmp;
73     }
74 }

```

3.2 Classe Colis (8 points) ***/**/****

```

75 package ds4eti2018_2S.modele;
76
77 import java.util.ArrayList;
78 import java.util.List;
79 import ds4eti2018_2S.modele.comportements.ComportementColis;
80
81 public class Colis implements ComportementColis{
82
83 //un attribut à trouver pour gérer l'incréméntation automatique des numéros de colis
84
85     private List<ComportementColis> caracteristiques = new
86     ArrayList<ComportementColis>();
87     private int numero;

```

```

88 //le constructeur ne prends pas de numéro : à chaque création de Colis, le numéro
89 est obtenu en ajoutant 1 au numéro du colis précédemment créé
90 public Colis(List<ComportementColis> caracteristiques){
91
92
93
94
95
96 }
97 public boolean hasComportement(ComportementColis cc) {
98 //renvoie vrai si le colis dispose du comportement cc
99
100
101
102
103
104 }
105
106 public List<ComportementColis> getCaracteristiques() {
107     return caracteristiques;
108 }
109
110 public String toString(){
111     String info="";
112     for(ComportementColis cc:caracteristiques){info+=cc+",";}
113     return "Colis n°"+numero+" (" +info+")";
114 }
115
116 @Override
117 public double getPrix(int km) {
118     double ret=0;
119     for(ComportementColis c:caracteristiques)
120     {
121         ret+=c.getPrix(km);
122     }
123     return ret*km;
124 }
125
126 @Override
127 public int getDelai() {
128     int ret=0;
129     for(ComportementColis c:caracteristiques)
130     {
131         if(ret>0 && c.getDelai()<ret){ret = c.getDelai();}
132         else{ret = c.getDelai();}
133     }
134     return ret;
135 }
136
137 }
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

3.3 MoyenTransport (3 points) **

```
131 package ds4eti2018_2S.modele.transport;
132
133 import java.util.ArrayList;
134 import java.util.List;
135
136 import ds4eti2018_2S.modele.Colis;
137 import ds4eti2018_2S.modele.comportements.ComportementColis;
138
139 public class MoyenTransport {
140     private List<ComportementColis> comportementsGeres = new
141 ArrayList<ComportementColis>();
142     private String immatriculation;
143     private List<Colis> chargement = new ArrayList<Colis>();
144
145     public MoyenTransport(String immatriculation, List<ComportementColis> cmp){
146         this.immatriculation = immatriculation;
147         comportementsGeres = cmp;
148     }
149
150     //ajoute le colis uniquement si le moyen de transport a bien TOUS les
151 comportements permettant de gérer le colis.
152     public boolean addColis(Colis c){
153         boolean ret = false;
154
155         return ret;
156     }
157
158     public List<ComportementColis> getComportementsGeres(){return
159 comportementsGeres;}
160
161     public String toString(){
162         String ret = "Chargement du véhicule "+immatriculation;
163         for(Colis c:chargement){
164             ret+="\n\t"+c;
165         }
166         return ret;
167     }
168 }
```

3.4 Comportements (3 points) *

```
169 package ds4eti2018_2S.modele.comportements;
170
171 public interface ComportementColis {
172     public double getPrix(int km);
173     public int getDelai();
174 }
175
176 package ds4eti2018_2S.modele.comportements;
177
178 public abstract class ComportementConcret implements ComportementColis,
179 Comparable<ComportementColis> {
180
181     public boolean equals(Object o){
```

```

183         return this.getClass().equals(o.getClass());
184     }
185     public int hashCode(){
186         return this.getClass().getSimpleName().hashCode();
187     }
188
189     public int compareTo(ComportementColis c){
190 //les comportements sont comparés en fonction de la comparaison du nom de leur
191 classe. Deux instances de comportement ayant la même classe sont égaux lorsqu'on les
192 compare.

```

```

193     }
194
195     public String toString(){return this.getClass().getSimpleName();}
196 }
197

```

```

198
199 package ds4eti2018_2S.modele.comportements;
200
201 public class CmpEco extends ComportementConcret {
202
203     @Override
204     public double getPrix(int km) {
205         return km*0.1;
206     }
207
208     @Override
209     public int getDelai() {
210         return 3;
211     }
212 }
213

```

```

214
215 package ds4eti2018_2S.modele.comportements;
216
217 public class CmpFroid extends ComportementConcret {
218
219     @Override
220     public double getPrix(int km) {
221 //prix = km*0,25 + 1€ tous les 100km

```

```

222     }
223
224     @Override
225     public int getDelai() {
226         return 2;
227     }
228 }
229
230

```



```

231 package ds4eti2018_2S.modele.comportements;
232
233 public class CmpUrgent extends ComportementConcret {
234
235     private static final double prixZ1 = 0.5;
236     private static final double prixZ2 = 1;
237     private static final int zone1 = 30;
238
239     @Override
240     public double getPrix(int km) {
241         double prix = 0;
242         //le prix dépend de la zone : si km est inférieur à zone1, le prix = prixZ1*km.
243         sinon, c'est prixZ2*km

```

```

244
245     }
246
247     @Override
248     public int getDelai() {
249         return 1;
250     }
251 }
252

```

3.5 Classe de Test (4 points) **

```

253 package ds4eti2018_2S.modele;
254 import java.util.ArrayList;
255 import java.util.List;
256
257 import ds4eti2018_2S.modele.comportements.CmpEco;
258 import ds4eti2018_2S.modele.comportements.CmpFroid;
259 import ds4eti2018_2S.modele.comportements.CmpUrgent;
260 import ds4eti2018_2S.modele.comportements.ComportementColis;
261 import ds4eti2018_2S.modele.transport.MoyenTransport;
262
263 public class Test {
264
265     public static List<Colis> fabriqueColis(int nb){
266         List<Colis> liste = new ArrayList<Colis>();
267         for (int i=0;i<nb;i++){
268             //50 % de chance d'ajouter un comportement Urgent
269             //50 % de chance AUSSI d'ajouter un comportement Froid
270             //100 % de chance d'ajouter le comportement Eco

```

```

271         }
272         return liste;}

```

```

273 public static void main(String[] args) {
274     ComportementColis c1 = new CmpUrgent();
275     ComportementColis c2 = new CmpFroid();
276     ComportementColis c3 = new CmpEco();
277
278     List<ComportementColis> lT1=new ArrayList<ComportementColis>();
279     List<ComportementColis> lT2=new ArrayList<ComportementColis>();
280     List<ComportementColis> lT3=new ArrayList<ComportementColis>();
281
282     lT1.add(c1);
283     lT1.add(c3);
284     lT2.add(c3);
285     lT3.add(c1);
286     lT3.add(c3);
287     List<MoyenTransport> transports = new ArrayList<MoyenTransport>();
288     transports.add(new MoyenTransport("1234FZ21",lT1));
289     transports.add(new MoyenTransport("4578AZ12",lT2));
290     transports.add(new MoyenTransport("BF214",lT3));
291
292     Agence a = new Agence("Mon Agence","Lyon", transports);
293     System.out.println(a);
294
295     List<Colis> l = fabriqueColis(20);
296     for(Colis c:l){
297         if(a.addColis(c)){System.out.println("Colis "+c +" ajouté");}
298         else{System.out.println("Colis "+ c + "non pris en charge");}
299     }
300
301     a.affecteChargement();
302     System.out.println("Prix du transport pour 25km: "+a.calculPrix(25));
303     System.out.println("Prix du transport pour 185km: "+a.calculPrix(185));
304
305 }
306
307 }

```

3.6 Exemple de Sortie de Test

```

308 Agence Mon Agence située à Lyon
309     CmpEco
310     CmpUrgent
311
312 Colis Colis n°1 (CmpUrgent,CmpFroid,CmpEco,)non pris en charge
313 Colis Colis n°2 (CmpFroid,CmpEco,)non pris en charge
314 Colis Colis n°3 (CmpUrgent,CmpEco,) ajouté
315 Colis Colis n°4 (CmpUrgent,CmpFroid,CmpEco,)non pris en charge
316 Colis Colis n°5 (CmpEco,) ajouté
317 Colis Colis n°6 (CmpFroid,CmpEco,)non pris en charge
318 Colis Colis n°7 (CmpUrgent,CmpFroid,CmpEco,)non pris en charge
319 Colis Colis n°8 (CmpFroid,CmpEco,)non pris en charge
320 Colis Colis n°9 (CmpUrgent,CmpEco,) ajouté
321 Colis Colis n°10 (CmpUrgent,CmpFroid,CmpEco,)non pris en charge
322 Colis Colis n°11 (CmpUrgent,CmpEco,) ajouté
323 Colis Colis n°12 (CmpEco,) ajouté
324 Colis Colis n°13 (CmpFroid,CmpEco,)non pris en charge
325 Colis Colis n°14 (CmpUrgent,CmpEco,) ajouté
326 Colis Colis n°15 (CmpFroid,CmpEco,)non pris en charge
327 Colis Colis n°16 (CmpUrgent,CmpEco,) ajouté
328 Colis Colis n°17 (CmpEco,) ajouté
329 Colis Colis n°18 (CmpUrgent,CmpFroid,CmpEco,)non pris en charge
330 Colis Colis n°19 (CmpUrgent,CmpFroid,CmpEco,)non pris en charge
331 Colis Colis n°20 (CmpUrgent,CmpFroid,CmpEco,)non pris en charge
332 Prix du transport pour 25km: 2062.5
333 Prix du transport pour 185km: 198505.0

```

4 Interface graphique (15 points)

4.1 (6 points) Proposez une maquette (dessin) d'interface graphique SIMPLE permettant :

- 1) d'ajouter un colis et ses caractéristiques
- 2) d'affecter les colis à des transports valables
- 3) d'afficher la liste des colis répartis dans les transports

Vous indiquerez le nom des composants swing que vous utilisez, ainsi que le fonctionnement des gestionnaires de positionnement.

4.2 Écrivez les parties de code impliquant des Listener (ajout, implémentation) (9 points)

5 Évolutions (8 points)

5.1 Comment prendre en compte la notion de capacité maximale pour un moyen de transport (par exemple, pas plus de 12 colis pour M1, 8 pour M2, etc.) (4 points)

5.2 On souhaite ajouter une fonctionnalité permettant de dresser le liste des colis qui n'ont pas été affecté par l'agence à des transports, faute de compétence pour ce type de colis. Expliquez comment prendre en compte cette fonctionnalité (4 points)