Titre du TIPE sur Jeux et Sports

Code du TIPE

Ceci est un modèle pour générer un PDF qui présente le code d'un TIPE.
Le fichier principal `rendu_code.tex` est à compiler avec l'option `-shell-escape`.

Conseils :

- Commencez par faire du ménage dans votre code, supprimer les fichiers obsolètes, les brouillons... Si ce n'est pas fait, c'est peut-être aussi l'occasion de séparer son code en plusieurs fichiers...

- Placez le dossier avec ce code latex à proximité de votre code dossier de code (pas dedans) et indiquez les fichiers de code à inclure grâce à un chemin relatif. Typiquement si votre dossier `TIPE` contient les dossiers `Code` et `rendu_code`, vous écrirez `../code/sous_dossier/fichier.c`. Il est déconseillé de copier tout son code dans un dossier spécial pour générer le rendu de code, cela n'est pas très robuste aux modifications et ajouts de code ultérieurs.

- Si vous importez le code du fichier en entier, pensez à supprimer les lignes vides en fin de fichier, les portions de code commenté devenu inutile...

- Si au contraire vous utilisez l'import par blocs, il est conseillé de sauter des lignes entre les différentes fonctions de sorte que chaque bloc commence à une ligne multiple de 10 (par exemple). Cela permet, dans le cas où vous modifiez a posteriori une fonction, en ajoutant une ligne de commentaire par exemple, d'avoir à modifier seulement les numéros de lignes du bloc correspondant, et pas ceux de tous les blocs suivants.

- Pensez à ajuster le modèle pour une version finale :
  · renseignez nom, prénom et numéro de candidat dans le fichier `mise_en_forme_MPI.tex`, pour la première page l.6 ET pour le pied de page l.32 ;
  · adaptez le titre et la date ;
  · supprimez cette section de conseils en commentant la l.20 ;
  · à la fin il ne doit plus y avoir de rouge dans le PDF.

- Compilez plusieurs fois si nécessaire pour ajuster le nombre total de pages en pied de page.

- Si vous ne réduisez pas la police, vous pouvez imprimer ce PDF livret (A4-> A5, en 2 pages par feuille...), ça reste lisible.

1 Code sur la première partie

1.1 Code du fichier

Ici, on insère le fichier entier, tel quel, pas forcément adapté.

```c
#include <stdio.h>
#include <stdbool.h>
#include <assert.h>

void type_simple_triangle (int a, int b, int c){
//hyp : a>=0 && b>=0 && c>=0
  if (a==b){
    if(b==c)
      printf("équilatéral");
    else
      printf("isocèle");
  }
  else{
    if((b==c) || (a==c))
      printf("isocèle");
    else
      printf("scalène");
  }
  printf("\n");
}




bool est_triangle(int a, int b, int c){
  return (a<=b+c) && (b<=a+c) && (c<=a+b) ;
}









bool est_plat(int a, int b, int c){
  return (a==b+c) || (b==a+c) || (c==a+b) ;
}









bool est_rectangle(int a, int b, int c){
  return (a*a + b*b == c*c) || (b*b + c*c == a*a) || (a*a + c*c == b*b) ;
}
```

```c
void affiche_type_triangle(int a,int b, int c){
  if (!est_triangle(a,b,c))
    printf("Ces longueurs ne sont les 3 côtés d'un même triangle\n");
  else{
    if (a==b){
      if(b==c){
         if (a==0)// alors a=b=c=0
           printf("équilatéral plat = réduit à un point");
         else
          printf("équilatéral");
        }
      else if(2*b*b == c*c)  // dans ce cas a²+b²=c²
        printf("isocèle rectangle");
      else if (a+b == c)
        printf("isocèle plat");
      else
        printf("isocèle");
    }
    else{ // ici on sait que a!=b
      if (b == c) {
        if(2*b*b == a*a)  // dans ce cas c²+b²=a²
          printf("isocèle rectangle");
        else if (b+c == a)
          printf("isocèle plat");
        else
          printf("isocèle");
      }
      else if (a == c){
        // ici on sait que a!=b et b!=c et a=c
        if(2*a*a == b*b)  // dans ce cas a²+c²=b²
          printf("isocèle rectangle");
        else if (a+c == b)
          printf("isocèle plat");
        else
          printf("isocèle");
      }
      else{ // ici on sait que a!=b et b!c et a!=c
        if (est_plat(a,b,c))
          printf("plat");
        else if (est_rectangle(a,b,c))
          printf("rectangle");
        else
          printf("scalène");
      }
    }
```

```
100     printf("\n");
101   }
102 }
103
104
105 int main(){
106   // jeu de test de type_triangle
107   type_simple_triangle (1,1,1); //equi
108   type_simple_triangle (1,1,2); //iso
109   type_simple_triangle (1,2,1); //iso
110   type_simple_triangle (2,1,1); //iso
111   type_simple_triangle (1,2,3); //scalene
112
113
114
115   // jeu de test de est_triangle
116   assert(est_triangle (1,2,3));
117   assert(est_triangle (3,4,5));
118   assert(est_triangle (1,1,1));
119   assert(!est_triangle (3,4,15));
120   assert(!est_triangle (31,4,15));
121
122
123
124
125   // jeu de test de est_plat
126   assert(est_plat (1,2,3));
127   assert(!est_plat (3,4,5));
128
129
130   // jeu de test de est_rectangle
131   assert(!est_rectangle (1,2,3));
132   assert(est_rectangle (3,4,5));
133
134
135   // jeu de test de affiche_type_triangle
136   affiche_type_triangle (1,1,1); //equi
137   affiche_type_triangle (0,0,0); // equi plat = pt
138   affiche_type_triangle (1,1,2); //iso plat
139   affiche_type_triangle (1,2,1); //iso plat
140   affiche_type_triangle (2,1,1); //iso plat
141   affiche_type_triangle (3,3,5); //iso
142   affiche_type_triangle (1,2,3); //plat
143   affiche_type_triangle (3,4,5); //rectangle
144   affiche_type_triangle (3,4,6); //scalène
145   //isocèle rectangle n'arrive pas avec des longueurs entières
146
147   return 0;
148 }
```

1.2 Code du fichier `nom_fichier.c`

Ici, on insère le fichier par bloc, de telle ligne à telle ligne. On utilise même `\textbackslash newpage` pour éviter que le bloc du `main` soit à cheval sur deux pages.

```c
#include <stdio.h>
#include <stdbool.h>
#include <assert.h>

void type_simple_triangle (int a, int b, int c){
//hyp : a>=0 && b>=0 && c>=0
  if (a==b){
    if(b==c)
      printf("équilatéral");
    else
      printf("isocèle");
  }
  else{
    if((b==c) || (a==c))
      printf("isocèle");
    else
      printf("scalène");
  }
  printf("\n");
}

bool est_triangle(int a, int b, int c){
  return (a<=b+c) && (b<=a+c) && (c<=a+b) ;
}

bool est_plat(int a, int b, int c){
  return (a==b+c) || (b==a+c) || (c==a+b) ;
}

void affiche_type_triangle(int a,int b, int c){
  if (!est_triangle(a,b,c))
    printf("Ces longueurs ne sont les 3 côtés d'un même triangle\n");
  else{
    if (a==b){
      if(b==c){
         if (a==0)// alors a=b=c=0
           printf("équilatéral plat = réduit à un point");
         else
          printf("équilatéral");
        }
      else if(2*b*b == c*c)  // dans ce cas a²+b²=c²
        printf("isocèle rectangle");
      else if (a+b == c)
        printf("isocèle plat");
      else
        printf("isocèle");
    }
    else{ // ici on sait que a!=b
```

```
74        if (b == c) {
75          if(2*b*b == a*a)  // dans ce cas c²+b²=a²
76            printf("isocèle rectangle");
77          else if (b+c == a)
78            printf("isocèle plat");
79          else
80            printf("isocèle");
81        }
82        else if (a == c){
83          // ici on sait que a!=b et b!=c et a=c
84          if(2*a*a == b*b)  // dans ce cas a²+c²=b²
85            printf("isocèle rectangle");
86          else if (a+c == b)
87            printf("isocèle plat");
88          else
89            printf("isocèle");
90        }
91        else{ // ici on sait que a!=b et b!c et a!=c
92          if (est_plat(a,b,c))
93            printf("plat");
94          else if (est_rectangle(a,b,c))
95            printf("rectangle");
96          else
97            printf("scalène");
98        }
99      }
100     printf("\n");
101   }
102 }
```

```c
int main(){
  // jeu de test de type_triangle
  type_simple_triangle (1,1,1); //equi
  type_simple_triangle (1,1,2); //iso
  type_simple_triangle (1,2,1); //iso
  type_simple_triangle (2,1,1); //iso
  type_simple_triangle (1,2,3); //scalene



  // jeu de test de est_triangle
  assert(est_triangle (1,2,3));
  assert(est_triangle (3,4,5));
  assert(est_triangle (1,1,1));
  assert(!est_triangle (3,4,15));
  assert(!est_triangle (31,4,15));




  // jeu de test de est_plat
  assert(est_plat (1,2,3));
  assert(!est_plat (3,4,5));


  // jeu de test de est_rectangle
  assert(!est_rectangle (1,2,3));
  assert(est_rectangle (3,4,5));


  // jeu de test de affiche_type_triangle
  affiche_type_triangle (1,1,1); //equi
  affiche_type_triangle (0,0,0); // equi plat = pt
  affiche_type_triangle (1,1,2); //iso plat
  affiche_type_triangle (1,2,1); //iso plat
  affiche_type_triangle (2,1,1); //iso plat
  affiche_type_triangle (3,3,5); //iso
  affiche_type_triangle (1,2,3); //plat
  affiche_type_triangle (3,4,5); //rectangle
  affiche_type_triangle (3,4,6); //scalène
  //isocèle rectangle n'arrive pas avec des longueurs entières

  return 0;
}
```

2 Code de la deuxième partie

2.1 Code du fichier