

# Document Globale

## SAE S1.01

# Sommaire

Description de l'application.....	2
Répartition des tâches.....	3
Graphe d'appel.....	4
Jeux d'essais.....	5
- Jeux d'essais du sous-programme Nbporte() .....	6
- Jeux d'essais du sous-programme CreerTableau().....	7
- Jeux d'essais du sous-programme NbErreur().....	8
- Jeux d'essais du sous-programme AfficherTemps().....	9
- Jeux d'essais du sous-programme ConversionTemps().....	10
- Jeux d'essais du sous-programme Tempfinal() .....	11
- Jeux d'essais du sous-programme SaisieNtc().....	12
- Jeux d'essais du sous-programme SaisieNtc() .....	13
- Jeux d'essais du sous-programme SaisieTableau().....	14
- Jeux d'essais du sous-programme Classement().....	15
- Jeux d'essais du sous-programme AfficherPodium() .....	16
- Jeux d'essais du sous-programme AfficherPodium().....	17
Code.....	18
Description de l'application.....	19

# Description de l'application

L'application a pour but d'assister les arbitres lors des compétitions de canoë en leur permettant de saisir et de traiter les résultats des compétiteurs.

Elle offre les fonctionnalités suivantes :

- Saisie des informations des compétiteurs.
- Enregistrement des résultats de chaque manche.
- Calcul automatique des temps compensés.
- Présentation du podium des trois meilleurs temps.

# Répartition des tâches

- Bonard Louis
  - Calcul des temps compensés pour l'affichage
- Drapied Hugo
  - Sous programmes saisie portes, erreurs , conversion du temps et la création du tableau principal
- Dupont Alexis
  - Réalisation et affichage du podium en minutes et secondes , et millisecondes
- Thamié Tinaël
  - saisir les valeurs des participants dans le tableau

# Graphe d'appel

```
public class CourseCanoë {
    public static void main(String args[]){
        Scanner clavier = new Scanner(System.in);
        String reponse="";
        int NbPorte=Nbporte();
        int Tableau[][];
        Tableau=CreerTableau();
        saisieTableau(Tableau, NbPorte);
        TempsFinal(Tableau);
        AfficherTemps(Tableau);
        int[][] TableauClassement;
        TableauClassement= CreerTabClass();
        Classement(Tableau,TableauClassement);
        AfficherPodium(TableauClassement);
        System.out.println("Voulez vous le temps de chaque participants ayant fini la course et le podium en minutes et secondes ? (oui/non)");
        reponse=clavier.nextLine();
        switch(reponse){
            case "oui" : {
                ConversionTemps(Tableau);
                PodiumMinSec(TableauClassement);
            }
        }
    }
}
```

## Jeux d'essais

- Jeux d'essais du sous-programme Nbporte() réalisé par **Drapied Hugo**

Porte	Résultat attendu
19	19
17	X
23	X

Résultat :

```
BlueJ: BlueJ : Terminal - SAE_DEV
Options
Combien de portes comporte votre course ?
17
Incorrect, votre parcours doit comporter entre 18 et 22 portes
23
Incorrect, votre parcours doit comporter entre 18 et 22 portes
19
```

Explication :

On contrôle que le nombre de portes dans le parcours soit entre 18 et 22

- Jeux d'essais du sous-programme CreerTableau() réalisé par **Drapied Hugo**

Joueur	Résultat attendu
8	tab = [0,0,0,0,0,0,0,0]
50	X
1	X

Résultat :

```
Rentrez le nombre de participants, il doit y en avoir entre 2 et 49.
50
Rentrez le nombre de participants, il doit y en avoir entre 2 et 49.
1
Rentrez le nombre de participants, il doit y en avoir entre 2 et 49.
8
[[I@15b4be6e
```

Explication :

On contrôle que le nombre de joueurs est bien compris entre 2 et 49

- Jeux d'essais du sous-programme NbErreur() réalisé par **Drapied Hugo**

pfPorte	Touché	Oublié	Résultat attendu
18	6	1	62
18	19	2	X
18	2	19	X

Résultat :

```

BlueJ: BlueJ : Terminal - SAE_DEV-copy
Options
Combien de portes le joueur a-t-il touché ?
19
Combien de portes le joueur a-t-il oublié ?
2
Le joueur ne peut pas faire plus d'erreur qu'il n'y a de portes, recommencez :
Combien de portes le joueur a-t-il touché ?
2
Combien de portes le joueur a-t-il oublié ?
19
Le joueur ne peut pas faire plus d'erreur qu'il n'y a de portes, recommencez :
Combien de portes le joueur a-t-il touché ?
6
Combien de portes le joueur a-t-il oublié ?
1
62

```

Explication :

On retourne le nombre de point d'erreur si le nombre d'erreur ne dépasse pas le nombre de porte



- Jeux d'essais du sous-programme AfficherTemps() réalisé par **Drapied Hugo**

Tableau[i][7]	Résultat attendu
0	Le participant n°"+PfTableau[i][0]+" n'a pas fini la course.
23000	Le participant n°"+PfTableau[i][0]+" a fait un temps de "+PfTableau[i][7]+" millisecondes.

Résultat :

Le participant n°1 n'a pas fini la course.

Le participant n°2 a fait un temps de 23000 millisecondes

Explication :

On parcourt le tableau, pour chaque colonne on vérifie si le temps composé est égal à 0 ou pas, si il l'est on renvoie que le joueur n'a pas fini la course. Sinon on renvoie que le temps du joueur en millisecondes.

- Jeux d'essais du sous-programme ConversionTemps() réalisé par **Drapied Hugo**

Tableau[i][7]	Résultat attendu
0	X
23000	Le participant n°1 a fait un temps de 0 minutes et 23 secondes


Explication :

On convertit le temps final du participant en temps sous le format minutes et secondes

- Jeux d'essais du sous-programme Tempfinal() réalisé par **Bonard Louis**

pfTab	Résultat attendu
{0, 1, 10000, 5, 1, 8000, 3, 0}, {1, 1, 12000, 6, 1, 9000, 4, 0}	26 000 millisecondes 31 000 millisecondes

Résultat :

 BlueJ: BlueJ : Terminal - SAE S1.01

Options

26000.0

31000.0

Explication : La manche 1 et 2 ont été validée donc le temps final va être l'addition des deux manche

pfTab	Résultat attendu
{0, 0, 10000, 5, 1, 8000, 3, 0},	0

Résultat :

 BlueJ: BlueJ : Terminal - SAE S1.01

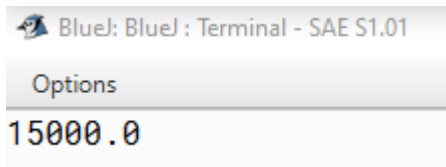
Options

0.0

Explication : La manche 1 n'a pas été validée donc le temps final sera de 0 vu qu'il a ratée sa première manche

pfTab	Résultat attendu
{0, 1, 10000, 5, 0, 8000, 3, 0},	15000 millisecondes

Résultat :



Explication : La manche 2 n'a pas été validée mais la manche 1 a été validée donc le temps final sera celui de la manche 1 vu qu'il a raté sa deuxième manche.

- Jeux d'essais du sous-programme SaisieNtc() réalisé par **Thamié Tinaël**

valeur	pfBorneinf	pfBornesup	Résultat attendu
0	0	50	0
10	0	50	10
51	0	50	X

résultat:

```
Donnez une valeur comprise entre 0 et 50.  
0  
0  
Donnez une valeur comprise entre 0 et 50.  
61  
Erreur ! Donnez une valeur comprise entre 0 et 50.  
10  
10
```

Explication :

On rentre les bornes et on rentre une valeur, cette valeur doit se retrouver entre les bornes

- Jeux d'essais du sous-programme SaisieNtc() réalisé par **Thamié Tinaël**

valeur	pfBorneinf	pfBornesup	Résultat attendu
0	0	50	0
10	0	50	10
51	0	50	X

résultat:

```
Donnez une valeur comprise entre 0 et 50.
0
0
Donnez une valeur comprise entre 0 et 50.
61
Erreur ! Donnez une valeur comprise entre 0 et 50.
10
10
```

Explication :

On rentre les bornes et on rentre une valeur, cette valeur doit se retrouver entre les bornes

- Jeux d'essais du sous-programme SaisieTableau() réalisé par **Thamié Tinaël**

Exemple 1 : Un joueur saisit "oui" pour la validation de la manche 1, entre un temps de 100000 ms et 2 erreurs. Puis il valide la manche 2 avec un temps de 200000 ms et 1 erreur. Cela permet de vérifier le fonctionnement du tableau pour les deux manches.

Exemple 2 : Un autre joueur ne valide pas la manche 1, on vérifie que la saisie pour la manche 2 n'est pas demandée. Ces jeux d'essais couvrent les cas où les manches sont validées ou non, permettant de vérifier que les conditions sont respectées.

- Jeux d'essais du sous-programme Classement() réalisé par **Dupont Alexis**

pftab	Résultat attendu
[0,0,0,0,0,0,0,0,21000]	pftabc = [1,19000]
[1,0,0,0,0,0,0,0,19000]	[0,21000]
[2,0,0,0,0,0,0,0,22000]	[2,22000]

Explication :

On classe les candidat du plus petit au plus gros temps

- Jeux d'essais du sous-programmer AfficherPodium() réalisé par **Dupont Alexis**

pftabc	Résultat attendu
[1,19000]	La première place revient au numéro 1 avec un temps de 19000 millisecondes
[0,21000]	La deuxième place revient au numéro 0 avec un temps de 21000 millisecondes
[2,22000]	La troisième place revient au numéro 2 avec un temps de 22000 millisecondes

Explication :

On affiche le podium de la course en millisecondes

- Jeux d'essais du sous-programme AfficherPodium() réalisé par **Dupont Alexis**

pftabc	Résultat attendu
[1,19000] [0,21000] [2,22000]	La première place revient au numéro 1 avec un temps de 0 minutes et 19 secondes  La deuxième place revient au numéro 0 avec un temps de 0 minutes et 21 secondes  La troisième place revient au numéro 2 avec un temps de 0 minutes et 22 secondes

Explication :

On affiche le podium de la course en minutes et secondes



# Code

```
public class CourseCanoe {
    public static void main(String args[]){
        Scanner clavier = new Scanner(System.in);
        String reponse="";
        int NbPorte=Nbporte();
        int Tableau[][];
        Tableau=CreerTableau();
        saisieTableau(Tableau, NbPorte);
        TempsFinal(Tableau);
        AfficherTemps(Tableau);
        int[][] TableauClassement;
        TableauClassement= CreerTabClass();
        Classement(Tableau,TableauClassement);
        AfficherPodium(TableauClassement);
        System.out.println("Voulez vous le temps de chaque participants ayant fini la course et le podium en minutes et secondes ? (oui/non)");
        reponse=clavier.nextLine();
        switch(reponse){
            case "oui": {
                ConversionTemps(Tableau);
                PodiumMinSec(TableauClassement);
            }
        }
    }
}

/**
 * @author Drapied Hugo
 */
public static int Nbporte(){
    int Porte=0;
    Scanner clavier = new Scanner(System.in);
    System.out.println("Combien de portes comporte votre course ?");
    Porte = clavier.nextInt();
    while (Porte <18 || Porte >22){ // On contrôle que le nombre de portes dans le parcours est entre 18 et 22
        System.out.println("Incorrect, votre parcours doit comporter entre 18 et 22 portes.");
        Porte = clavier.nextInt();
    }
    return Porte;
}

/**
 * @author Drapied Hugo
 */
public static int[][] CreerTableau() {
    int tableau[][];
    Scanner clavier = new Scanner(System.in);
    int joueur=0;
    while (joueur >=50 || joueur <=1){ //On contrôle que le nombre de joueurs est bien compris entre 2 et 49
        System.out.println("Rentrez le nombre de participants, il doit y en avoir entre 2 et 49.");
        joueur = clavier.nextInt();
    }
    tableau = new int[joueur][8];
    return tableau;
}

/**
 * @author Drapied Hugo
 */
public static int NbErreur(int pfPorte){
    Scanner clavier = new Scanner(System.in);
    int Toucher=0;
    int Oubli=0;
    int Somme=-1;
    while (Somme>pfPorte || Somme<0){ //On contrôle que le nombre d'erreurs entre 0 et pfPorte (nombre de portes du parcours)
        System.out.println("Combien de portes le joueur a-t-il touché ?");
        Toucher= clavier.nextInt();
        System.out.println("Combien de portes le joueur a-t-il oublié ?");
        Oubli=clavier.nextInt();
        Somme=Toucher+Oubli;
        if (Somme > pfPorte){
            System.out.println("Le joueur ne peut pas faire plus d'erreur qu'il n'y a de portes, recommencez : \n");
        }
    }
}
```

```

/**
 * @author Hugo
 */
public static void AfficherTemps(int[][] PfTableau){
    /*Scanner clavier = new Scanner(System.in);
    int[][] PfTableau = new int [2][8];
    for (int i=0; i<2;i++){
        for (int j=0; j<8; j++){
            PfTableau[i][j]=clavier.nextInt();
        }
    }*/
    int joueur = PfTableau.length;
    /* On parcourt le tableau, pour chaque colonne on vérifie si le temps composé est égal à 0 ou pas, si il l'est on renvoie que le joueur tab[i][j=0]
    n'a pas fini la course. Sinon on renvoie que le joueur tab[i][j=0] a fait pour temps tab[i][j=7] */
    for (int i=0; i<joueur; i++){
        if (PfTableau[i][7]==0){
            System.out.println("Le participant n°"+PfTableau[i][0]+" n'a pas fini la course.");
        } else {
            System.out.println("Le participant n°"+PfTableau[i][0]+" a fait un temps de "+PfTableau[i][7]+" millisecondes.");
        }
    }
    System.out.println("");
}

```

```

/**
 * @author Drapiéd Hugo
 */
public static void ConversionTemps(int[][] PfTableau) {
    /*Scanner clavier = new Scanner(System.in);
    int[][] PfTableau = new int [2][8];
    for (int i=0; i<2;i++){
        for (int j=0; j<8; j++){
            PfTableau[i][j]=clavier.nextInt();
        }
    }*/
    int joueur = PfTableau.length;
    int minute=0;
    int seconde=0;
    for (int i=0; i<joueur; i++){
        if (PfTableau[i][7]!=0){
            minute = (PfTableau[i][7]/1000)/60;
            seconde = (PfTableau[i][7]/1000)%60;
            System.out.println("Le participant n°"+PfTableau[i][0]+" a fait un temps de "+minute+"m et "+seconde+"s.");
        }
    }
    System.out.println("");
}

```

```

import java.util.Scanner;
public class Course
{
    /**
     * Calcule le temps final pour chaque participant basé sur les données des manches.
     *
     * @param pfTab Tableau 2D contenant les informations des participants.
     * @return Tableau mis à jour avec les temps finaux pour chaque participant.
     */
    public static double[][] TempsFinal(double [][]pfTab){// tableau a 8 ligne
        int tempfinal = 0; // déclaration de la variable temps
        int i; // compteur et colonne de la table
        int j; // compteur et ligne de la table

        for(i=0; i < pfTab.length; i++){// boucle for i < nbr de participant
            pfTab[i][3] *= 1000;
            pfTab[i][6] *= 1000;
            if (pfTab[i][1] == 1){//vérification manche 1 réussi, 1 = true
                for (j=2; j < 4; j++){ // temps de la manche1 + pénalité manche 1
                    tempfinal += pfTab[i][j];
                }
            }
            if(pfTab[i][4] == 1){//vérification manche 2 réussi, 1 = true
                for (j=5; j < 7; j++){// temps de la manche 2 + pénalité manche 2
                    tempfinal += pfTab[i][j];
                }
            }
            pfTab[i][7] = tempfinal;
            //System.out.println(pfTab[i][7]);
            tempfinal = 0;
        }

        //System.out.println(tempfinal);
        //System.out.println(pfTab.length);
        return pfTab; //retourner le temp final
    }
}

```

```

/**
 * @author Thamié Tinaël
 */
public static int saisieIntC(int pfBorneInf, int pfBorneSup){
    int valeur ;
    Scanner clavier = new Scanner(System.in) ;
    System.out.println("Donnez une valeur comprise entre "+pfBorneInf+" et "+pfBorneSup+" .") ;
    valeur = clavier.nextInt() ;
    while (valeur<pfBorneInf || valeur>pfBorneSup){
        System.out.println("Erreur ! Donnez une valeur comprise entre "+pfBorneInf+" et "+pfBorneSup+" .") ;
        valeur = clavier.nextInt() ;
    }
    System.out.println(valeur);
    return valeur;
}

```

```

public static String saisie(){
    Scanner clavier = new Scanner(System.in) ;
    String saisie = clavier.nextLine() ;
    return saisie ;
}

```

```

/**
 * @author Tamié Thinaël
 */
public static void saisieTableau(int[][] pfTab, int pfnbPortes) {
    for (int i = 0 ; i < pfTab.length ; i++) {
        pfTab[i][0] = i + 1;

        System.out.println("Le joueur numéro "+pfTab[i][0]+" a-t-il validé la manche 1 ? (si oui : saisissez oui ou o)");
        String saisie = saisie() ;
        switch(saisie){
            case "oui": {
                pfTab[i][1] = 1;

                System.out.println("Veuillez saisir le temps (en ms) parcouru à la première manche :)");
                pfTab[i][2] = saisieIntC(0, 240000);

                pfTab[i][3] = NbErreur(pfnbPortes);
            }
        }
    }

    for (int i = 0 ; i < pfTab.length ; i++) {
        if (pfTab[i][1] == 1) {

            System.out.println("Le joueur numéro "+pfTab[i][0]+" a-t-il validé la manche 2 ? (si oui : saisissez oui ou o)");
            String saisie = saisie() ;
            switch(saisie){
                case "oui": {
                    pfTab[i][4] = 1;

                    System.out.println("Veuillez saisir le temps (en ms) parcouru à la deuxième manche :)");
                    pfTab[i][5] = saisieIntC(0, 240000);

                    pfTab[i][6] = NbErreur(pfnbPortes);
                }
            }
        }
    }
}

```

```

/**
 * @author Dupont Alexis
 */
public static int[][] CreerTabClass(){
    int[][] TabClass;
    TabClass = new int [3][2];
    return TabClass;
}

```

```

* @author Dupont Alexis
*/
public static void Classement(int[][] pfTab, int[][] pfTabC){
    int joueurs= pfTab.length;
    int premier= 0;           //numéro de brassard du premier
    int deuxieme= 0;          //numéro de brassard du second
    int troisieme= 0;          //numéro de brassard du troisième
    int tempsP=1000000000;     // temps compensé du premier
    int tempsD=1000000000;     // temps compensé du deuxième
    int tempsT=1000000000;     // temps compensé du troisième
    pfTabC[0][0]=0;
    pfTabC[1][0]=0;

    for (int i=0; i<joueurs; i++){
        if (tempsP>pfTab[i][7] && pfTab[i][7]!=0){
            tempsP=pfTab[i][7];
            premier=pfTab[i][0];
        }
    }
    pfTabC[0][0]=premier;
    pfTabC[0][1]=tempsP;
    for (int i=0; i<joueurs; i++){
        if (tempsD>pfTab[i][7] && pfTab[i][0]!=premier && pfTab[i][7]!=0){
            tempsD=pfTab[i][7];
            deuxieme=pfTab[i][0];
        }
    }
    pfTabC[1][0]=deuxieme;
    pfTabC[1][1]=tempsD;

    // Recherche du troisième temps
    if (joueurs>=3){
        for (int i=0; i<joueurs; i++){
            if (tempsT>pfTab[i][7] && pfTab[i][0]!=premier && pfTab[i][0]!=deuxieme && pfTab[i][7]!=0){
                tempsT=pfTab[i][7];
                troisieme=pfTab[i][0];
            }
        }
        pfTabC[2][0]=troisieme;
        pfTabC[2][1]=tempsT;
    } else {
        pfTabC[2][0]=0;
    }
}

```

```

/**
 * @author Dupont Alexis
 */
public static void AfficherPodium(int[][] pftabC){
    if (pftabC[0][0]==0){ // On vérifie si personne n'a fini la course (pas de premier)
        System.out.println("Personne n'a fini la course, il n'y a pas de vainqueurs\n");
    } else if (pftabC[1][0]==0 && pftabC[0][0]!=0){ // On vérifie qu'au moins une personne ait fini la course (1 premier)
        System.out.println("Le vainqueur est "+pftabC[0][0]+" avec un temps de "+pftabC[0][1]+" il n'y a pas de second ou troisième\n");
    } else if (pftabC[2][0]==0 && pftabC[1][0]!=0 && pftabC[0][0]!=0 && pftabC[1][1]==pftabC[0][1]){ // On vérifie si seulement deux personnes ont fini la course en faisant le même temps(2)
        System.out.println("Les numéros "+pftabC[0][0]+" et "+pftabC[1][0]+" sont premiers ex aequo avec un temps de "+pftabC[0][1]+".");
    } else if (pftabC[2][0]==0 && pftabC[1][0]!=0 && pftabC[0][0]!=0 && pftabC[1][1]!=pftabC[0][1]){ // On vérifie si seulement deux personnes ont fini la course en en faisant pas le même temps(2)
        System.out.println("La première place revient au numéro "+pftabC[0][0]+" avec un temps de "+pftabC[0][1]+" millisecondes.\n");
        System.out.println("La deuxième place revient au numéro "+pftabC[1][0]+" avec un temps de "+pftabC[1][1]+" millisecondes.\n");
    } else if (pftabC[0][1]==pftabC[1][1] && pftabC[1][1]==pftabC[2][1]){ // On vérifie si les trois meilleurs ont fait le même temps (3 premiers)
        System.out.println("Les numéros "+pftabC[0][0]+", "+pftabC[1][0]+", "+pftabC[2][0]+" sont premiers ex aequo avec un temps de "+pftabC[0][1]+" millisecondes.\n");
    } else if (pftabC[0][1]==pftabC[1][1] && pftabC[1][1]!=pftabC[2][1]){ //On vérifie si le premier et deuxième sont ex aequo (2 premiers et 1 troisième)
        System.out.println("Les numéros "+pftabC[0][0]+" et "+pftabC[1][0]+" sont premiers ex aequo avec un temps de "+pftabC[0][1]+".\n");
        System.out.println("La troisième place revient au numéro "+pftabC[2][0]+" avec un temps de "+pftabC[2][1]+" millisecondes.\n");
    } else if (pftabC[0][1]!=pftabC[1][1] && pftabC[1][1]==pftabC[2][1]){ //On vérifie si le deuxième et troisième sont ex aequo (1 premier et 2 deuxièmes)
        System.out.println("La première place revient au numéro "+pftabC[0][0]+" avec un temps de "+pftabC[0][1]+" millisecondes.\n");
        System.out.println("Les numéros "+pftabC[1][0]+" et "+pftabC[2][0]+" sont deuxièmes ex aequo avec un temps de "+pftabC[1][1]+".\n");
    } else if (pftabC[0][1]!=pftabC[1][1] && pftabC[1][1]!=pftabC[2][1]){ //On vérifie si personne n'a fait le même temps
        System.out.println("La première place revient au numéro "+pftabC[0][0]+" avec un temps de "+pftabC[0][1]+" millisecondes.\n");
        System.out.println("La deuxième place revient au numéro "+pftabC[1][0]+" avec un temps de "+pftabC[1][1]+" millisecondes.\n");
        System.out.println("La troisième place revient au numéro "+pftabC[2][0]+" avec un temps de "+pftabC[2][1]+" millisecondes.\n");
    }
}

```

```

/**
 * @author Dupont Alexis
 */
public static void PodiumMinSec(int[][] pftabC){
    int Minutes1=(pftabC[0][1]/1000)/60;
    int Secondes1=(pftabC[0][1]/1000)%60;
    int Minutes2=(pftabC[1][1]/1000)/60;
    int Secondes2=(pftabC[1][1]/1000)%60;
    int Minutes3=(pftabC[2][1]/1000)/60;
    int Secondes3=(pftabC[2][1]/1000)%60;
    if (pftabC[0][0]==0){ // On vérifie si personne n'a fini la course (pas de premier)
        System.out.println("Personne n'a fini la course, il n'y a pas de vainqueurs");
    } else if (pftabC[1][0]==0 && pftabC[0][0]!=0){ // On vérifie qu'au moins une personne ait fini la course (1 premier)
        System.out.println("Le vainqueur est "+pftabC[0][0]+" avec un temps de "+Minutes1+"min et "+Secondes1+"sec il n'y a pas de second ou troisième");
    } else if (pftabC[2][0]==0 && pftabC[1][0]!=0 && pftabC[0][0]!=0 && pftabC[1][1]==pftabC[0][1]){ // On vérifie si seulement deux personnes ont fini la course en faisant le même temps(2)
        System.out.println("Les numéros "+pftabC[0][0]+" et "+pftabC[1][0]+" sont premiers ex aequo avec un temps de "+Minutes1+"min et "+Secondes1+"sec.");
    } else if (pftabC[2][0]==0 && pftabC[1][0]!=0 && pftabC[0][0]!=0 && pftabC[1][1]!=pftabC[0][1]){ // On vérifie si seulement deux personnes ont fini la course en en faisant pas le même temps(2)
        System.out.println("La première place revient au numéro "+pftabC[0][0]+" avec un temps de "+Minutes1+"min et "+Secondes1+"sec .");
        System.out.println("La deuxième place revient au numéro "+pftabC[1][0]+" avec un temps de "+Minutes2+"min et "+Secondes2+"sec .");
    } else if (pftabC[0][1]==pftabC[1][1] && pftabC[1][1]==pftabC[2][1]){ // On vérifie si les trois meilleurs ont fait le même temps (3 premiers)
        System.out.println("Les numéros "+pftabC[0][0]+", "+pftabC[1][0]+", "+pftabC[2][0]+" sont premiers ex aequo avec un temps de "+Minutes1+"min et "+Secondes1+"sec .");
    } else if (pftabC[0][1]==pftabC[1][1] && pftabC[1][1]!=pftabC[2][1]){ //On vérifie si le premier et deuxième sont ex aequo (2 premiers et 1 troisième)
        System.out.println("Les numéros "+pftabC[0][0]+" et "+pftabC[1][0]+" sont premiers ex aequo avec un temps de "+Minutes1+"min et "+Secondes1+"sec.");
        System.out.println("La troisième place revient au numéro "+pftabC[2][0]+" avec un temps de "+Minutes3+"min "+Secondes3+"sec.");
    } else if (pftabC[0][1]!=pftabC[1][1] && pftabC[1][1]==pftabC[2][1]){ //On vérifie si le deuxième et troisième sont ex aequo (1 premier et 2 deuxièmes)
        System.out.println("La première place revient au numéro "+pftabC[0][0]+" avec un temps de "+Minutes1+"min et "+Secondes1+"sec .");
        System.out.println("Les numéros "+pftabC[1][0]+" et "+pftabC[2][0]+" sont deuxièmes ex aequo avec un temps de "+Minutes2+"min et "+Secondes2+"sec.");
    } else if (pftabC[0][1]!=pftabC[1][1] && pftabC[1][1]!=pftabC[2][1]){ //On vérifie si personne n'a fait le même temps
        System.out.println("La première place revient au numéro "+pftabC[0][0]+" avec un temps de "+Minutes1+"min et "+Secondes1+"sec .");
        System.out.println("La deuxième place revient au numéro "+pftabC[1][0]+" avec un temps de "+Minutes2+"min et "+Secondes2+"sec .");
        System.out.println("La troisième place revient au numéro "+pftabC[2][0]+" avec un temps de "+Minutes3+"min et "+Secondes3+"sec .");
    }
}

```