```php
<?php
/*******************************************************************************
AUTEUR      : Constantin Herrmann
LIEU        : CFPT Informatique Genève
DATE        : 14.06.2018

TITRE PROJET: KidsGames Geneva Score

TITRE PAGE  : points_counting.c
DESCRIPTION : Ce script contient toutes les fonctions concernant le calcule des points pour les matchs
VERSION     : 1.0
*******************************************************************************/

/**
* Met à jour les scores et le classement général
*
* @param array un tableau contenant les infos des équipes ainsi que les scores obtenus lors du match
* @param string l'id du match dans lequel les scores on été obtenus
*/
function UpdateScore($Teams, $idGame){

  // Va rechercher les infos du match
  $game = GetAllMatchInfos($idGame);
  // Nous récupérons les résultats du match avant la mise à jour des nouveaux résultats
  $old_results = GetMatchResults($idGame, $game);

  // On indique que le match est terminé
  SetGameToDone($idGame);
  $game = GetAllMatchInfos($idGame);

  // On parcours le tableau des équipes
  for ($team=0; $team < count($Teams); $team++) {

    // On garde l'id de l'équipe dans une variable a part
    $idTeam = $Teams[$team]['infos']['numero'];
    // Idem que pour l'id mais pour le score
    $score = $Teams[$team]['score'];

    // On éffectue la mise a jour du score dans la base de données
    // ATTENTION : On ne met pas la variable $new_score mais bien la variable $score
    static $query = null;

    if ($query == null) {
      $req = "UPDATE `plays` SET `score`= :score WHERE `idTeam` = :idTeam AND `idGame` = :idGame";
      $query = connecteur()->prepare($req);
    }
    try {
      $query->bindParam(':idTeam', $idTeam, PDO::PARAM_STR);
      $query->bindParam(':idGame', $idGame, PDO::PARAM_STR);
      $query->bindParam(':score', $score, PDO::PARAM_STR);
      $query->execute();
      $query->fetch();
    }
```

```
    catch (Exception $e) {
      error_log($e->getMessage());
    }
  }


  // Nous récupérons les résultats du match après la mise a jour des nouveaux
  résultats
  $new_results = GetMatchResults($idGame, $game);

  /* On recherche de quel classement il s'agit
  Tchouckball = B
  KinBall = A
  Course Agile = C */
  $classment = GetTypeOfClassement($idGame);
  $c_label = null;
  if ($classment['sport'] == 1) {
    $c_label = 'B';
  }elseif ($classment['sport'] == 2) {
    $c_label = 'A';
  }elseif ($classment['sport'] == 3) {
    $c_label = 'C';
  }

  // On parcours toutes les équipes du match
  for ($team=0; $team < count($Teams); $team++) {
    // On garde l'id de l'équipe dans une variable a part
    $idTeam = $Teams[$team]['infos']['numero'];
    // Idem que pour l'id mais pour le score
    $score = $Teams[$team]['score'];

    // On déclare 2 variables provisoires représentant le vieux score et le
    nouveau
    // par défaut on les met a NULL
    $old_provisoire = null;
    $new_provisoire = null;

    // on parcours tout les anciens résultats
    for ($res=0; $res < count($old_results); $res++) {
      // Si l'id du détenteur du résultat est égal a l'id de l'équipe dans la
      boucle
      if ($old_results[$res]['id'] == $idTeam) {
        // On assigne le score à la variable provisoire : old
        $old_provisoire = $old_results[$res];
        break;
      }
    }

    // Idem que la boucle ci-dessus, sauf que cette fois-ci on le fait pour le
    nouveau score
    for ($res=0; $res < count($new_results); $res++) {
      if ($new_results[$res]['id'] == $idTeam) {
        $new_provisoire = $new_results[$res];
        break;
      }
    }

    // On va récuperer les données du classement général des équipes
```

```
108     $main = GetMainClassement($idTeam, $c_label);
109 
110     // Création du tableau qui contiendra les nouvelles données à entrer
111     $final_new_results = [];
112 
113     /*
114     On calcule le nombres de points obtenus grace a la formule : points =
    •   pointsactuels + (nouveaux_points - anciens_points)
115 
116     SI C'EST UN MODIFICATION DE SCORES ERRONES :
117     Ex:
118       Points actuels = 6;
119       nouveaux_points = 1; -> L'équipe à fait égalité avec une autre
120       anciens_points = 3; -> L'équipe avait gagné le match, mais c'etait des
    •   mauvais résultats
121 
122       points = 6 + ( 1 - 3 ) = 4
123 
124     SI C'EST UN NOUVEAU MATCH :
125     Ex:
126       Points actuels = 6;
127       nouveaux_points = 3; -> L'équipe à gagné son match
128       anciens_points = 0; -> L'équipe n'avait pas encore joué le match
129 
130       points = 6 + ( 3 - 0 ) = 9
131 
132 
133     On fait idem pour les points marqués et les points reçus
134     */
135     $final_new_results['p'] = $main['p'] + ($new_provisoire['points'] -
    •   $old_provisoire['points']);
136     $final_new_results['m'] = $main['m'] + ($new_provisoire['m'] -
    •   $old_provisoire['m']);
137     $final_new_results['r'] = $main['r'] + ($new_provisoire['r'] -
    •   $old_provisoire['r']);
138 
139     // On met à jour la table principale
140     UpdateMainTable($idTeam, $final_new_results['p'], $final_new_results['m'],
    •   $final_new_results['r'], $c_label);
141   }
142 }
143 
144 /**
145 * Retourne un tableau avec les points et les score par équipe pour un match en
    •   question
146 *
147 * @param string l'id du match
148 * @param array le match en question
149 * @return array un tableau avec tout les scores ainsi que les poitns obtenus par
    •   équipe
150 *           [index]
151 *              ['id'] -> l'id de l'équipe
152 *              ['m'] -> points marqués par l'équipe
153 *              ['r'] -> nombre de points encaisser
154 *              ['d'] -> la différence de points
155 *              ['points'] -> les points recus grâce au match (3 si victoire, 1 si
    •   égalité, 0 si défaite)
156 */
157 function GetMatchResults($idGame, $game){
```

```php
function getMatchResults($idGame, $game){
  static $query = null;

  if ($query == null) {
    $req = "SELECT `idTeam` as id, `score` FROM `plays` WHERE `idGame` = :idGame
    ORDER BY `score` DESC";
    $query = connecteur()->prepare($req);
  }

  try {
    $query->bindParam(':idGame', $idGame, PDO::PARAM_STR);
    $query->execute();
    $res = $query->fetchAll(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  $result = $res;
  // On va créer les résultats pour le match
  $points = MakeTheResults($result, $game);
  return $points;
}

/**
* Retourne un tableau avec les points et les score par équipe pour un match en
question
*
* @param array les équipes qui ont participer au match
* @param array le match en question
* @return array un tableau avec tout les points obtenus par équipe
*           [index]
*              ['id'] -> l'id de l'équipe
*              ['m'] -> points marqués par l'équipe
*              ['r'] -> nombre de points encaisser
*              ['d'] -> la différence de points
*              ['points'] -> les points recus grâce au match (3 si victoire, 1 si
égalité, 0 si défaite)
*/
function MakeTheResults($teams, $game){
  $result = [];

  // On parcours toutes les équipes
  for ($temp=0; $temp < count($teams); $temp++) {
    $result[$temp]['id'] = $teams[$temp]['id'];
    $result[$temp]['m'] = $teams[$temp]['score'];
    $result[$temp]['r'] = 0;
    $result[$temp]['d'] = $teams[$temp]['score'];
    $result[$temp]['points'] = 0;
    for ($r=0; $r < count($teams); $r++) {
      if ($r != $temp) {
        $result[$temp]['d'] = $result[$temp]['d'] - $teams[$r]['score'];
        $result[$temp]['r'] = $result[$temp]['r'] + $teams[$r]['score'];
      }
    }
  }
  // Création des points d'après leur position dans le match
  $result = MakethePoints($result, $game);
```

```php
  return $result;
}

/**
* Caclucule les points pour chaque équipe
*
* @param array les équipes classées par résultats obtenus lors du match
* @param array le match en question
* @return array un tableau avec tout les points pour chaque équipe
*/
function MakethePoints($teams, $game){
  $table = $teams;

  // Si il y a 2 équipes (Tchouckball)
  if (count($teams) == 2) {
    // On récupère les points marqués par les équipes
    $t1 = $teams[0]['m'];
    $t2 = $teams[1]['m'];

    // Si les score de la première équipe est plus grand que celui de la deuxième
    if ($t1 > $t2) {
      // Le gagnant reçois 3 points et le perdant 0
      $table[0]['points'] = 3;
      $table[1]['points'] = 0;
    }
    // Si les deux scores sont à égalité
    elseif ($t1 == $t2) {
      // On vérifie si c'est un nouveau match ou si il à déjà été joué (donc
      MODIFICATION)
      if ($game['infos']['played'] == "1") {
        $table[0]['points'] = 1;
        $table[1]['points'] = 1;
      }
      else{
        $table[0]['points'] = 0;
        $table[1]['points'] = 0;
      }
    }
    // Si le score de la première équipe est plus petit que le score de la
    deuxième
    else{
      // Le gagnant reçois 3 points et le perdant 0
      $table[0]['points'] = 0;
      $table[1]['points'] = 3;
    }

  }
  // Si il y a 3 équipes (KinBall et course agile)
  elseif(count($teams) == 3){
    $t1 = $teams[0]['m'];
    $t2 = $teams[1]['m'];
    $t3 = $teams[2]['m'];
    // Si le score de l'équipe 1 est plus grand que tout les autres scores
    if ($t1 > $t2 && $t1 > $t3) {
      // Alors l'équipe 1 reçois 3 points
      $table[0]['points'] = 3;

      // Si le score de l'équipe 2 est plus grand que celui de l'équipe 3
```

```php
      if ($t2 > $t3) {
        // L'équipe 2 reçois 1 points
        $table[1]['points'] = 1;
        // L'équipe 3 reçois 0 points
        $table[2]['points'] = 0;
      }
      // L'équipe 2 et l'équipe 3 ont les mêmes scores
      elseif ($t2 == $t3) {
        // Les 2 équipes recoivent 1 point
        $table[1]['points'] = 1;
        $table[2]['points'] = 1;
      }
    }
    // Si les deux prmières équipes sont égalités ET que la troisième équipe a un score inferieur aux deux autres
    elseif ($t1 == $t2 && $t1 > $t3) {
      // Les deux premières équipes recoivent 1 point
      $table[0]['points'] = 1;
      $table[1]['points'] = 1;

      // La troisième équipe reçois aucun point
      $table[2]['points'] = 0;
    }
    // Si toutes les équipes sont à égalités
    elseif ($t1 == $t2 && $t1 == $t3) {
      // Si le match à déjà été joué
      if ($game['infos']['played'] == "1") {
        // Toutes les équipes reçoivent 1 point
        $table[0]['points'] = 1;
        $table[1]['points'] = 1;
        $table[2]['points'] = 1;
      }

      // Si le match n'as pas encore été joué
      else{
        // Toutes les équipes reçoivent aucun point
        $table[0]['points'] = 0;
        $table[1]['points'] = 0;
        $table[2]['points'] = 0;
      }
    }
  }

  return $table;
}

/**
* Met à jour la table du classement principale
*
* @param string id de l'équipe
* @param string les points à ajouter
* @param string les points marqués
* @param string les points reçus
* @param string le classement dans lequel ajouter les points
*/
function UpdateMainTable($idTeam, $points_to_add, $points_m, $points_r, $classement){
  try {
```

```php
    $req = "UPDATE `Teams` SET `p_$classement` = $points_to_add,`m_$classement` =
    $points_m,`r_$classement` = $points_r WHERE `id` = $idTeam";
    $query = connecteur()->prepare($req);
    $query->execute();
    $query->fetch();
  }
  catch (Exception $e) {
    error_log($e->getMessage());
  }
}

/**
* Retorune l'id du Sport du match
*
* @param string id du match
* @return array un tableau avec l'id du match
*           ['idSport'] -> l'id du sport
*/
function GetTypeOfClassement($idGame){
  static $query = null;

  if ($query == null) {
    $req = "SELECT `idSport` as sport FROM `Games` WHERE `id` = :idGame";
    $query = connecteur()->prepare($req);
  }
  try {
    $query->bindParam(':idGame', $idGame, PDO::PARAM_STR);
    $query->execute();
    $res = $query->fetch(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }
  return $res;
}

/**
* Met à jour le champ "played" de la tables games
* Met le champ à 1, ce qui indique que le match à été joué
*
* @param string id du match (game)
*/
function SetGameToDone($id){
  static $query = null;

  if ($query == null) {
    $req = "UPDATE `Games` SET `played`= 1 WHERE `id` = :id";
    $query = connecteur()->prepare($req);
  }
  try {
    $query->bindParam(':id', $id, PDO::PARAM_STR);
    $query->execute();
    $query->fetch();
  }
  catch (Exception $e) {
    error_log($e->getMessage());
  }
```

```php
}

/**
* Retourne le Score pour une équipe dans un match (game)
*
* @param string id de l'équipe
* @param string id du match (game)
* @return array un tableau avec le scores
*               ['score'] -> le score de l'équipe dans le match
*/
function GetScoreForTeamInMatch($idTeam, $idGame){
  static $query = null;

  if ($query == null) {
    $req = "SELECT `score` FROM `plays` WHERE `idTeam` = :idTeam AND `idGame` =
    :idGame";
    $query = connecteur()->prepare($req);
  }
  try {
    $query->bindParam(':idTeam', $idTeam, PDO::PARAM_STR);
    $query->bindParam(':idGame', $idGame, PDO::PARAM_STR);
    $query->execute();
    $res = $query->fetch(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }
  return $res;
}

```