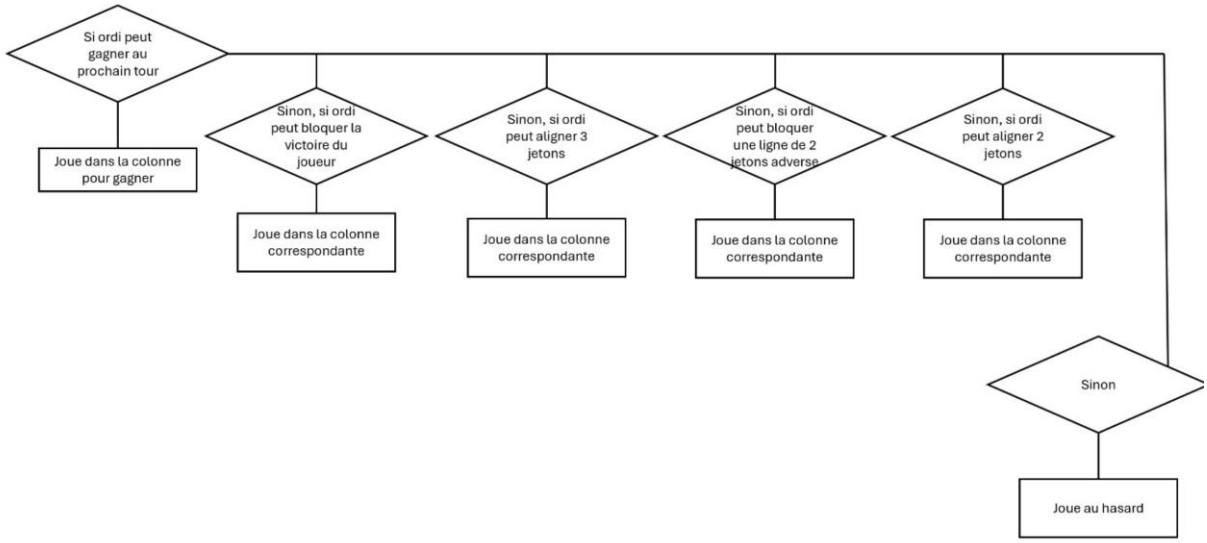
Projet : Puissance 4 Contre la Machine

Contre-rendu 4

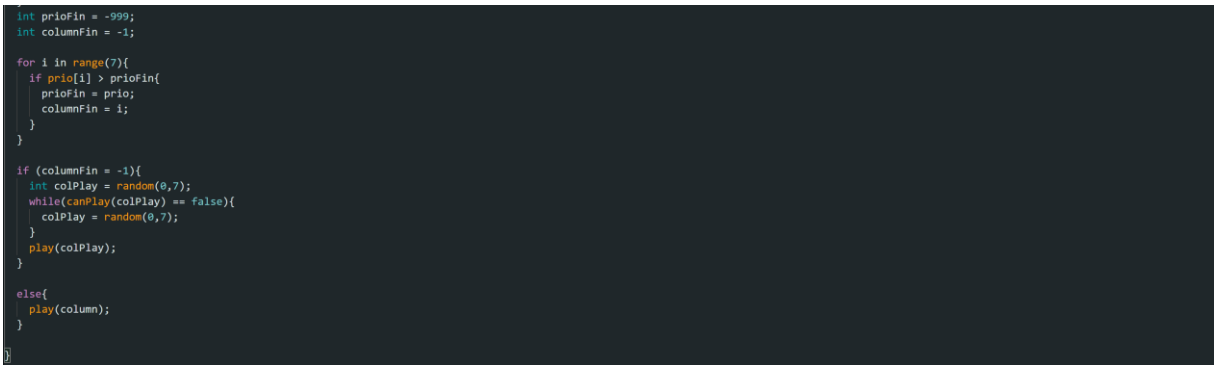
Tout comme je le pressentais, je n'ai pas réussi à finaliser l'algorithme et si je ne fais pas de changement cela risque de de compromettre le projet. J'ai donc changé d'algorithme :



Cet algorithme va simplement tester de jouer un jeton dans les 7 colonnes une à une puis les soumettre à un certain nombre de tests avant de classer chaque coup possible selon son efficacité : il choisira ensuite le coup avec la priorité la plus haute, c'est-à-dire le plus avantageux. Sinon, si tous les coups ont la même priorité, il joue au hasard.
Cet algorithme est beaucoup moins puissant que le dernier puisque qu'il a une profondeur de test beaucoup plus limitée et ne va jouer que ce qu'il l'avantage immédiatement, mais il fera l'affaire pour mettre en œuvre notre projet.

Voici la fonction principale de ce nouvel algorithme (toutes les autres fonctions appelées dans celle-ci sont codées plus haut) :

```
void loop() {
  // put your main code here, to run repeatedly:
  prio = {0,0,0,0,0,0,0};
  for i in range(7){
    int histoTempoC[][2] = histoC;
    int columnsTempo[7] = columns;

    if (canPlay(i) == true){
      columnsTempo[i] += 1;
      int posTempo[2] = {i, columnsTempo[i]};
      histoTempoC[nbMoves(histoTempoC)] = posTempo;

      if (checkWin(posTempo, histoTempoC) == true){       //Si l'ordi peut gagner en jouant dans cette colonne, prio absolue
        prio[i] = 100;
      }

      else if (checkLine3(posTempo, histoJ) == true) || (checkColumn3(posTempo, histoJ)) || (checkDiago3(posTempo, histoJ))){ //Si l'ordi peut bloquer une victoire adverse
        prio[i] = 99;
      }

      else if (checkLine3(posTempo, histoTempoC) == true) || (checkColumn3(posTempo, histoTempoC) == true) || (checkDiago3(pos, histoTempoC)){ //Si l'ordi peut aligner trois jetons
        prio[i] = 10;
      }

      else if (checkLine2(posTempo, histoTempoJ) == true) || (checkColumn2(posTempo, histoTempoJ) == true) || (checkDiago2(pos, histoTempoJ)){ //Si le joueur peut aligner deux jetons
        prio[i] = 5;
      }
      else if (checkLine2(posTempo, histoTempoC) == true) || (checkColumn2(posTempo, histoTempoC) == true) || (checkDiago2(pos, histoTempoC)){  //Si l'ordi peut aligner deux jetons
        prio[i] = 3;
      }

    }

    else{
      prio = -1000;
    }
  }
```



```
  int prioFin = -999;
  int columnFin = -1;

  for i in range(7){
    if prio[i] > prioFin{
      prioFin = prio;
      columnFin = i;
    }
  }

  if (columnFin = -1){
    int colPlay = random(0,7);
    while(canPlay(colPlay) == false){
      colPlay = random(0,7);
    }
    play(colPlay);
  }

  else{
    play(column);
  }
}
```

J'ai assigné pour l'instant des valeur arbitraires pour les priorités, mais il suit l'algorithme décrit plus haut.

Je dois encore rajouter quelques fonctionnalités comme considérer le cas où, par exemple, l'ordinateur peut aligner 3 jetons en jouant dans une certaine colonne, mais que cela permettrait au joueur adverse de gagner, ou d'aligner 3 jetons etc... Cela rajoutera quelques situations supplémentaires à considérer mais cela rendra l'algorithme légèrement plus intelligent sans pour autant coûter trop de ressources supplémentaires.

Je finirais l'écriture et les tests du code ce week-end et espère l'avoir complété d'ici la prochaine séance.