C - Créateur de joueur de foot 🏆

Projet

5TTR 6TTR Découverte

Dans cet exercice, tu vas programmer un **CRÉATEUR DE JOUEUR** en langage C. Chaque joueur reçoit un **capital de points aléatoire**, réparti automatiquement selon son **poste** (gardien, défenseur, milieu ou attaquant). Tu devras ensuite permettre à l'utilisateur de **distribuer ces points** dans les différentes **caractéristiques du joueur**, puis calculer une **note globale** en fonction de ses performances et de son comportement.

Objectifs pédagogiques

Ce projet te permettra de mettre en pratique :

- les **variables** (types de base, structures),
- les **opérations arithmétiques** et les **conditions** (`if`, `else`),
- les **boucles** (`for`, `while`) pour les saisies répétées,
- les **fonctions** pour organiser ton programme,
- et la notion de **pondération** (répartition de points selon des pourcentages).

Principe du programme

1. L'utilisateur crée un joueur (nom, âge, numéro, poste).
2. Le programme **tire aléatoirement** un **budget total de points**, compris entre **60 % et 80 %** de 1200 points. → Ce total est donc compris entre **720 et 960 points**.
3. Ce budget est ensuite **réparti automatiquement** entre les **4 groupes d'attributs** (Physiques, Techniques, Tactiques/Mentaux et Généraux) selon le **poste choisi**.
4. L'utilisateur **distribue manuellement** les points de chaque groupe dans 4 caractéristiques, la **dernière** étant calculée automatiquement pour atteindre exactement le budget du groupe.
5. Enfin, le joueur entre ses **statistiques de saison** (buts, passes, arrêts, cartons, hors-jeu) et le programme calcule une **note finale sur 100**.

© 2025 Mr. Lambrechts

Pondérations par poste

Ces pondérations servent **au début** du programme pour déterminer le nombre de points alloués à chaque groupe d'attributs.

Poste	Physiques	Techniques	Tactiques	Généraux
1 – Gardien	20 %	15 %	25 %	40 %
2 – Défenseur	25 %	15 %	35 %	25 %
3 – Milieu	25 %	30 %	30 %	15 %
4 – Attaquant	20 %	40 %	25 %	15 %

Étape 1 — Informations générales

À demander :

- `nom` (texte, 30 caractères max)
- `age` (entier, entre 10 et 45)
- `numero` (entier, entre 1 et 99)
- `poste` (1 à 4)

Exemple d'affichage :

```
=== Créateur de joueur ===
Nom du joueur : Alex
Âge (10..45) : 23
Numéro de maillot : 9
Poste [1=Gardien, 2=Défenseur, 3=Milieu, 4=Attaquant] : 1
OK : Alex (#9), 23 ans, Poste = Gardien
```

Étape 2 — Tirage du budget total et répartition

Le programme tire un **budget total aléatoire** entre 720 et 960 points :

```
--- Budget total ---
Tirage aléatoire (60%..80% de 1200) → budget_total = 900 pts
```

© 2025 Mr. Lambrechts

Il répartit ensuite ce budget entre les 4 groupes selon le **poste** choisi. Exemple pour un **gardien** (20%, 15%, 25%, 40%) :

```
--- Répartition par pondérations ---
Physiques  : 20% → 180 pts
Techniques : 15% → 135 pts
Tactiques  : 25% → 225 pts
Généraux   : 40% → 360 pts
Vérification : 180 + 135 + 225 + 360 = 900 ✅
```

> ⚠️ Aucun groupe ne peut dépasser **400 POINTS** (car 4 caractéristiques × 100 max).

Étape 3 — Répartition manuelle dans chaque groupe

L'utilisateur distribue les points du groupe dans **3 attributs**, la **4e est calculée automatiquement** pour compléter le budget exact.

S'il dépasse le budget ou si la 4e valeur est hors limites (0..100), il doit recommencer la saisie du groupe.

Exemple 3A – Groupe *Physiques* (budget 180)

```
--- Attributs PHYSIQUES (budget: 180 pts) ---
vitesse (0..100) ? 70   [reste: 110]
endurance (0..100) ? 50 [reste: 60]
force (0..100) ? 30     [reste: 30]
--> agilite = 30
Valider ce groupe ? (Y/n) Y
```

Exemple 3B – Groupe *Techniques* (budget 135)

```
--- Attributs TECHNIQUES (budget: 135 pts) ---
précision (0..100) ? 50  [reste: 85]
contrôle (0..100) ? 40   [reste: 45]
dribble (0..100) ? 35    [reste: 10]
--> tir = 10
Valider ce groupe ? (Y/n) Y
```

Même logique pour les deux autres groupes :

- **Tactiques/Mentaux** : vision du jeu, défense, attaque, fair-play
- **Généraux** : placement, pressing, passe longue, duel aérien

---

Étape 4 — Statistiques du joueur

L'utilisateur encode les statistiques de saison :

```
--- Statistiques ---
Buts ? 3
Passes décisives ? 5
Arrêts ? 18
Cartons jaunes ? 1
Cartons rouges ? 0
Hors-jeu ? 2
```

Étape 5 — Calculs et note finale

1. Calcul de la **moyenne** de chaque groupe (sur 100).
2. Calcul d'un **score pondéré** selon le poste (moyenne des groupes × pondération).
3. Calcul d'un **score statistique** (selon le poste) :
   - Gardien : `5 × arrets`
   - Défenseur : `8 × buts + 5 × passes`
   - Milieu : `5 × buts + 8 × passes`
   - Attaquant : `10 × buts + 4 × passes` → Score limité à 100.
4. Calcul d'un **malus fair-play** :

   ```
   malus = 2 × cartons_jaunes + 8 × cartons_rouges + hors_jeu
   (capé à 30)
   ```

5. Calcul de la **note finale** :

   ```
   note = 0.7 × score_attributs + 0.3 × score_statistiques − malus
   (bornée entre 0 et 100)
   ```

Exemple de calcul affiché :

```
--- Calculs ---
Score attributs (pondéré) = 64.2
Score statistiques = 90
Malus fair-play = 4
Note finale = 67.9 / 100
```

Étape 6 — Fiche récapitulative

Affichage final attendu :

```
================ FICHE JOUEUR ================
Nom: Alex  (#9)   Âge: 23   Poste: Gardien
Budget total tiré : 900 pts
Budgets par groupe: Phys=180  Tech=135  Tact=225  Gen=360
-- PHYSIQUES : vit=70 end=50 for=30 agi=30   (moy=45.0)
-- TECHNIQUES: pré=50 ctr=40 dri=35 tir=10   (moy=33.8)
-- TACTIQUES : vis=60 def=55 att=60 fair=50  (moy=56.3)
-- GENERAUX  : pla=90 prs=85 pas=95 dua=90   (moy=90.0)
Stats: buts=3, passes=5, arrêts=18, CJ=1, CR=0, HJ=2
----------------------------------------------
Attributs pondérés : 64.2
Score statistiques  : 90
Malus fair-play     : 4
NOTE GLOBALE        : 67.9 / 100
Verdict : Excellent gardien sur sa ligne ; technique à perfectionner.
==============================================
```

Détails de programmation attendus

- Structure `Joueur` avec sous-structures pour chaque groupe d'attributs.
- Fonctions conseillées :

```c
int tirerBudgetTotal(void);
void repartirBudgetParPoste(int poste, int total, int* bPhys, int* bTech, int* bTact, int
void saisirGroupe(const char* nom, int budget, int* a, int* b, int* c, int* d);
double calculScoreAttributs(int poste, double phys, double tech, double tact, double gen)
double calculScoreStats(int poste, int buts, int passes, int arrets);
int calculMalus(int cj, int cr, int hj);
double calculNoteFinale(double attr, double stats, int malus);
void afficherFiche(const Joueur* j);
```

Critères d'évaluation

Élément	Points
Saisie + validations (groupes et bornes)	5
Gestion du budget global et des pondérations	4
Calculs corrects (scores, note finale, malus)	6
Code structuré et commenté	3
Affichage clair et complet	2

© 2025 Mr. Lambrechts

© 2025 Mr. Lambrechts https://v2.ttrinfo.be 6 | 6