# Tweede programmeeropdracht: Knock Out
# Algoritmiek voorjaar 2016, Universiteit Leiden

In deze opdracht schrijven we een programma om een geschikt toernooischema te maken voor de knock-out fase van de Champions League voetbal. In deze knock-out fase spelen 16 clubs in een toernooi, waarbij in iedere ronde de helft van de teams als winnaar van het veld stapt, totdat er maar één overblijft. Deze 16 clubs zijn de nummers 1 en 2 uit een voorafgaande groepsfase met acht groepen. Het schema moet aan de volgende drie criteria voldoen:

1. In de eerste knock-out ronde (de achtste finale) speelt altijd de nummer 1 van een groep tegen de nummer 2 van een andere groep.

2. De nummers 1 en 2 uit een zelfde groep mogen elkaar op z'n vroegst (als ze niet uitgeschakeld worden) pas in de finale weer tegenkomen.

3. Uit commercieel oogpunt vindt de UEFA (de organiserende voetbalbond) het bovendien prettig dat clubs uit hetzelfde land pas zo laat mogelijk tegen elkaar kunnen spelen. Als er twee clubs uit hetzelfde land zijn, mogen ze elkaar op z'n vroegst in de finale tegenkomen. Bij drie of vier clubs uit hetzelfde land op z'n vroegst in de halve finale, enzovoort. In theorie kunnen alle 16 clubs uit hetzelfde land afkomstig zijn[1]

Ontwerp een programma dat, gegeven een lijst van 16 deelnemers met bijbehorend land, een correct speelschema levert voor ronde 1, 2, 3, en de uiteindelijke finale. Je levert dus een lijst op die correspondeert met het plaatje in Figuur 1 op de volgende bladzijde, waarbij de clubs op de plaatsen A tot en met P ingevuld worden. Deze lijst (= het speelschema) moet voldoen aan de drie hierboven genoemde criteria van de UEFA. Daarnaast moet je programma ook het aantal verschillende correcte speelschema's bepalen. Als er geen enkel speelschema te maken is dat aan alle drie criteria van de UEFA voldoet, wordt de knock-out fase van het toernooi voor dit seizoen afgelast.

## Opdracht

De opdracht bestaat wederom uit twee delen: een C++-programma en een verslag.

1. Het C++-programma moet een klasse Toernooi implementeren.
   Op `http://www.liacs.leidenuniv.nl/~graafjmde/ALGO/` staat een skeletprogramma dat de structuur van deze klasse aangeeft. In principe hoef je alleen de bestanden `toernooi.h` en `toernooi.cc` aan te passen. De klasse moet in ieder geval de volgende drie memberfuncties hebben:

   **leesclubsin** Deze functie leest een tekstbestand in dat de clubs en de landen van deze clubs bevat.

   **bepaalschema** Deze functie vult op basis van de ingelezen clubs, de plaatsen A tot en met P in het speelschema (en daarmee het hele schema) in. Het resultaat wordt naar een ander tekstbestand geschreven. Als er een schema gevonden is dat aan alle criteria voldoet, kan deze functie stoppen.

[1] Dit is in de werkelijkheid van 2016 niet mogelijk, maar gezien de ontwikkelingen van de laatste jaren zou het in de toekomst mogelijk kunnen worden.


Figuur 1: Leeg speelschema voor knock-out fase Champions League.

**aantalschemas** Deze functie telt voor de ingelezen clubs het aantal verschillende correcte speelschema's. Het aflopen van alle mogelijkheden kan wel enkele minuten duren.

In het skeletprogramma is te zien welke parameters deze drie functies moeten hebben. Daarnaast voldoet je programma aan de volgende eisen:

- Het algoritme voor het bepalen van een speelschema (of alle correcte speelschema's) maakt gebruik van **backtracking**. Bedenk hoe je datastructuur eruit moet zien om de huidige situatie te representeren.
- Boven elke functie moet een commmentaarblokje komen met daarin een (zeer) korte beschrijving van wat de functie doet. Noem daarin ook de gebruikte parameters: geef hun betekenis en geef aan hoe ze eventueel veranderd worden door de functie. Geef bij memberfuncties ook aan wat deze met de membervariabelen van het object doen. Let ook op de layout (consequent inspringen) en op het overige commentaar bij de programmacode (alleen zinvol en kort commentaar).

Hint: het lijkt verstandig om na het inlezen van de invoer eerst te tellen hoeveel clubs er uit elk land meedoen. Dit kun je vervolgens gebruiken bij de implementatie van criterium 3 op pagina 1.

2. Het verslag moet getypt zijn in LaTeX, en moet een introductie bevatten, de probleemstelling, de gebruikte oplossingsmethode, een kleine analyse van het aantal mogelijkheden, en een hoofdstukje met resultaten.

   Bij de oplossingsmethode leg je onder andere duidelijk uit in welke volgorde je de clubs op welke plaatsen in het schema probeert te zetten. Hoe controleer je of een deeloplossing nog uit te breiden is tot een volledige oplossing? Betrek ook de gebruikte datastructuur in deze uitleg.

   Bij de analyse bereken je hoeveel correcte schema's er zijn in het speciale geval dat alle clubs uit verschillende landen komen. Zie je in dit geval symmetrieën in het speelschema, waardoor verschillende schema's feitelijk op hetzelfde neerkomen? Wat voor gevolgen heeft dit voor het aantal schema's? Bij de resultaten geef je voor vijf instanties (waaronder de drie instanties die meegeleverd worden bij het skeletprogramma) een resulterend schema en het aantal correcte schema's.

Invoer en uitvoer:

- Een invoerbestand voor deze programmeeropdracht bevat 16 regels, die elk bestaan uit een getal tussen 1 en 16, een spatie en een string van maximaal 25 letters uit het alfabet (hoofdletters en/of kleine letters) en/of cijfers. Het getal staat voor het land, de string is de naam van de club. Regels 1 en 2 bevatten respectievelijk de nummers 1 en 2 van groep A uit de groepsfase. Vervolgens komen de nummers 1 en 2 van groep B, enzovoort. Voor voorbeelden, zie de drie bestanden die meegeleverd worden bij het skeletprogramma.

- Het uitvoerbestand van de memberfunctie bepaalschema bevat opnieuw 16 regels, die er **op dezelfde manier** uitzien als de regels in de invoer: met het landnummer, een spatie en de naam van de club. Alleen nu staan ze in de volgorde waarin de clubs in het speelschema zijn terechtgekomen, op plaatsen A tot en met P. Als er geen schema gevonden kon worden, wordt dat in het uitvoerbestand gemeld.

Zie: `http://www.liacs.leidenuniv.nl/~graafjmde/ALGO/` voor eventuele aanvullingen of tips bij de programmeeropdracht. De behaalde cijfers komen t.z.t. op blackboard te staan.

**Uiterste (!) inleverdatum**: maandag 18 april 2016, uiterlijk 12:00 uur.
Het programma en `Makefile` per e-mail sturen naar: `j.m.de.graaf@liacs.leidenuniv.nl`. Zorg dat het onderwerp van je mail begint met "[ALGO]", dat scheelt de docent een hoop gezoek. Het verslag (inclusief het programma!) moet op papier worden ingeleverd in de daartoe bestemde doos met opschrift Algoritmiek in de postkamer van Informatica, kamer 156. Voor elke week te laat inleveren gaat er een punt van het cijfer af. Vermeld overal duidelijk de namen van de makers.

**Normering**: verslag 3 punten; commentaar en layout 1 punt; modulaire opbouw en OOP 1 punt; werking 5 punten.