Hertentamen Algoritmiek (I00009)

donderdag 22 februari 2007, 13:30 - 15:30 uur, HG00.068

- Deze herkansing is gesloten boek. Je mag geen literatuur of PC gebruiken.
- Het tentamen bestaat uit drie opgaven die in zwaarte verschillen. Aan het begin van iedere opdracht staat aangegeven hoeveel punten je maximaal kunt halen. Onderdelen worden eveneens verschillend gewaardeerd. Het aantal punten dat je maximaal per onderdeel kunt halen staat in de kantlijn vermeld.
- Ga niet overhaast te werk en lees de opgaven nauwkeurig. Begin met de opgaven die je het eenvoudigst lijken. Denk eerst goed na. Hou je antwoorden kort en controleer ze voor je het werk inlevert. Schroom niet vragen te stellen over eventuele onduidelijkheden.

1 Klassen en Rijen: Lingo **totaal 45 punten**

Lingo is een televisieprogramma. Hierbij gaat het om een woordspel dat een combinatie is van *Mastermind* en *Bingo*. De titel van het programma is waarschijnlijk afgeleid van het woord *letterbingo*.

De oorspronkelijk versie van het spel werd gespeeld met 2 teams. Daarbij kreeg elk team 5 pogingen om een 5-letterwoord te raden. Het team dat de beurt krijgt, krijgt de beginletter van het te raden woord en moet een woord beginnend met de gegeven beginletter noemen binnen een gegeven tijdslimiet. Dit woord dient een bestaand Nederlands woord te zijn en bovendien juist gespeld te worden. Vervolgens zal de computer aangeven welke letters op de juiste plek staan en welke letters niet op de juiste plek staan maar wel op een andere plek in het woord zitten. Deze *score* wordt per beurt aangegeven. Wanneer een team niet op tijd een woord weet te noemen of wanneer een woord verkeerd gespeld wordt, gaat de beurt naar het andere team. Dat team krijgt een bonusletter, een ingevulde letter op de eerste nog open plek in het woord.

Bij Lingo wordt slim spel gecombineerd met een gezonde dosis geluk, en deze formule is een succes gebleken. Het programma wordt al vanaf 1989 uitgezonden, en is daarmee het langstlopende spelletje op de Nederlandse televisie. In eerste instantie werd Lingo gespeeld met vijfletterwoorden, maar sinds oktober 2002 draait het om het raden van zesletterwoorden. Even dreigde het spelletje in 2007 definitief van de buis te verdwijnen, maar na veel protest, zelfs uit de Tweede Kamer, is dit besluit althans voorlopig van de baan.

In deze opdracht ga je Lingo implementeren in C++ waarbij echter niet in twee teams gespeeld wordt maar één speler het spel met de computer kan spelen. Het idee is dat de computer allereerst een geheim 5-letter woord bepaalt en de eerste letter van dat woord op het scherm laat zien. De speler probeert vervolgens het woord in maximaal 5 pogingen te raden. Na elke poging laat de computer de score van de speler zien. Deze score wordt op bovenbeschreven wijze bepaald en laat dus zien welke letters op de juiste plek staan en welke letters elders voorkomen.

Om je een eind op weg te helpen geven we het een en ander voor. Maar zoals altijd mag je dit naast je neerleggen en geheel je eigen gang gaan, mits je maar antwoord geeft op de gestelde vragen.

Voor het opslaan van 5-letter woorden gebruiken we een rij van tekens (**char**s). Dit leidt tot de volgende declaraties:

```
const int MaxAantalPogingen = 5;

const int AantalLetters = 5;

typedef char VijfLetterwoord [AantalLetters+1];
```

Voor letterscores introduceren we een enumeratietype en een procedure om zo'n score op het scherm te zetten. Als de score `LetterEnPlekOK` is dan laten we de eveneens meegegeven letter zien.

```
enum Score { Fout, LetterOK, LetterEnPlekOK };

void toonScore (Score s, char l)
{
    switch (s)
    {   case Fout:           cout << '-'; break;
        case LetterOK:       cout << '*'; break;
        case LetterEnPlekOK: cout << l; break;
    }
}
```

Het te raden woord houden we verborgen door het als **private** variabele in een klasse op te nemen. Dit leidt tot de volgende klassedefinitie.

```
class Lingo
{
private:
     VijfLetterwoord doel;

public:
    Lingo ();
    bool  letterStaatGoed (char letter, int pos);
    bool  letterKomtVoor (char letter);
    bool  woordIsGeraden (VijfLetterwoord poging);
    Score bepaalLetterScore (char letter, int pos);
    void  toonResultaat (VijfLetterwoord poging);
    void  vraagWoord (VijfLetterwoord poging);
    void  speel ();
};
```

De methodes uit deze klasse zijn deels al geïmplementeerd; de ontbrekende delen dien je straks zelf in te vullen.

```
Lingo :: Lingo ()
{   strcpy (doel, "appel"); }

bool Lingo :: letterStaatGoed (char letter, int pos)
{   return doel [pos] == letter; }

bool Lingo :: woordIsGeraden (VijfLetterwoord poging)
{   for (int i = 0; i < AantalLetters; i++)
```

```
        if (! letterStaatGoed (poging[i], i))
            return false;
    return true;
}

void Lingo :: vraagWoord (VijfLetterwoord poging)
{   for (int i = 0; i < AantalLetters; i++)
        cin.get (poging[i]);
    cin.get ();
}
```

Ook al zijn de voorgegeven methodes niet echt moeilijk geven we toch een korte uitleg. De constructor initialiseert het geheime doel-woord. We maken er ons hier met een jantje-van-leiden van af door steeds het woord `"appel"` in te vullen. De methode `letterStaatGoed` controleert of de opgegeven letter op de eveneens opgegeven positie in het doel-woord voorkomt. De methode `woordIsGeraden` gaat na of het geraden woord overeenkomt met het geheime woord. Met behulp van `vraagWoord` wordt de volgende poging van de speler ingelezen. Hierin wordt gebruik gemaakt van de ook op het college behandelde invoerfuntie `get`. Het maakt overigens niet veel uit of je de precieze werking van `vraagWoord` meteen ziet.

En dan nu de eigenlijke opdracht. Hierin zijn nog twee vereenvoudigingen aangebracht in vergelijking tot het oorspronkelijke spel. Het geraden woord hoeft niet getest te worden op spelling en correctheid en bovendien hoeft de beginletter niet dezelfde te zijn als die van het doel-woord. Implementeer nu de volgende operaties.

10 punten **1.a**) De methode `letterKomtVoor` die nagaat of de opgegeven letter ergens in het woord `doel` voorkomt.

10 punten **1.b**) De methode `bepaalLetterScore` die bepaalt wat de `Score` is van de letter `l` op positie `pos`. Dus controleer eerst of de letter `l` op positie `pos` staat. Is dat niet het geval dan controleer je of de letter ergens anders in het geheime woord voorkomt. **Hint**: maak gebruik van de twee vorige methodes.

15 punten **1.c**) De (wat ingewikkeldere) methode `toonResultaat` die van het als parameter meegegeven woord `poging` bepaalt wat de score van iedere letter is en vervolgens die score op het scherm laat zien. Voorbeeld: Stel het te raden woord is `"appel"` en de speler tikt `"aldus"` in. Op het scherm wordt de score van deze poging als volgt weergegeven:

a*——

De eerste letter is correct, de tweede letter staat niet op de juiste plek maar komt wel in het woord voor en de overige drie letters zijn alle verkeerd. **Hint**: Maak wederom handig gebruik van de vorige methodes.

5 punten **1.d**) Met de methode `speel` kan het spel worden gespeeld. Dat wil zeggen, de speler wordt net zo lang gevraagd een woord op te geven totdat hetzij het woord geraden is, hetzij het maximum aantal pogingen overschreden is. 'n Klein gedeelte van deze methode is al voorgegeven.

```
void Lingo :: speel ()
{
    VijfLetterwoord poging = "_____";
    poging[0] = doel[0];
    toonResultaat (poging);
```

```
        bool klaar = ! woordIsGeraden (poging);
        // onderdeel e
    }
```

Te beantwoorden vraag: "Waar dient dit voorgegeven deel voor?"

10 punten **1.e**) Completeer de body van `speel`

2 Rijen van getallen **totaal 25 punten**

10 punten **2.a**) Gegeven een rij van gehele getallen, met bijv. als inhoud: $\{1, 2, 3, 4, 5\}$. Maak een procedure `draaiom`, die na een aanroep als `draaiom(rij, 5);` de volgorde van de in de meegegeven rij aanwezige waarden omdraait (de gebruikte 2de parameter geeft de betreffende rijlengte aan). (Na aanroep van die procedure heeft de genoemde rij dus als inhoud: $\{5, 4, 3, 2, 1\}$.)

15 punten **2.b**) Gegeven nu 2 rijen van gehele getallen, waarbij beide rijen van dezelfde lengte zijn. Onder het [rij]product van twee rijen van gelijke lengte, verstaan we de som van de producten van de waarden in die 2 rijen, die op gelijke posities staan. Als bijvoorbeeld r_1 en r_2 de respectievelijke inhouden $\{1, 2, 3\}$ en $\{-2, 4, -6\}$ hebben, dan is hun rijproduct: $1 * -2 + 2 * 4 + 3 * -6 = -12$. Maak nu een functie `rijproduct` die bij aanroep beide rijen als parameter meekrijgt en bovendien als derde parameter de gemeenschappelijke `rijlengte`.

3 Recursie: wat doet deze functie? **totaal 15 punten**

Gegeven de volgende functie:

```
const char alphabet[] = "abcdefghijklmnopqrstuvwxyz" ;

void doeIetsMetTekst ( char regel[], int startwaarde )
{
    const int lengte = strlen(regel) ;
    if ( startwaarde > 0 && startwaarde < lengte )
    {   if ( startwaarde == (lengte / 2 ) )
            regel[startwaarde] = '\0' ;
        else
        {   if ( startwaarde % 2 == 0 )
                regel [startwaarde] = alphabet[startwaarde] ;
            doeIetsMetTekst ( regel, startwaarde + 1 ) ;
        }
    }
}
```

15 punten **3**) Geef aan wat het resultaat zal zijn van het volgende programmafragment en geef duidelijk aan, hoe je tot dat resultaat bent gekomen.

```
char regel[] = "0123456789ABCD" ;
cout << "\nDe oorspronkelijke tekst: " << regel ;
doeIetsMetTekst (regel, 2 ) ;
cout << "\nis omgezet in: " << regel << endl ;
```