Beschrijving van een experimenteel programma voor een leeromgeving

Instituut voor Perceptie Onderzoek
Postbus 513, 5600 MB Eindhoven

28.06.1988

Rapport no. 654

Beschrijving van een experimenteel programma voor een leeromgeving

T. Kolks
F. v. Rooy

Stagebegeleider: Drs. H. Ellermann
Stageplaats : IPO Eindhoven

Inhoudsopgave: blz.

Hfdst. 1: De probleemstelling.

1.1 Opdracht:

Voor onze stage is de volgende opdracht uitgereikt:
"Schrijf een programma dat in staat moet zijn om een lijst woorden, bijvoorbeeld in een vreemde taal, en de daarbij horende vertaling, aan een persoon aan te bieden, opdat de gebruiker die woorden met de vertaling kan leren. Hierbij staan bijvoorbeeld de vreemde woorden links op het computer-beeldscherm en de bijbehorende vertaling rechts op het scherm."
Het programma is bedoeld voor experimentele doeleinden en kan dus in de toekomst door derden worden aangepast.

1.2 Probleemspecificatie:

Het programma zal ontwikkeld moeten worden op en voor een SUN 3/50 systeem, in de taal C.
In het programma moeten de volgende parameters ingesteld kunnen worden:

1. Het aantal woorden dat links en rechts op het scherm staat.
2. De aard van de feedback waarmee een foutief antwoord gepaard gaat.
3. Het algoritme, waarmee de volgorde van de woorden die gevraagd worden bepaald wordt.
4. De tijd waarin een antwoord bestudeerd kan worden.

Bovendien moeten er een aantal terugmeldingen komen omtrent de vordering van de persoon achter de terminal, tot op een bepaald moment.

1.3 Aannamen:

Bij het maken van het programma hebben we aangenomen dat de te overhoren woorden uit niet meer dan 30 karakters bestaan. Bovendien mag de invoerfile niet meer dan 64 te overhoren woorden bevatten.
Voorts is aangenomen dat de proefpersoon niet meer dan negen uitvoerfiles op zijn naam krijgt te staan.

Hfdst. 2: Oplossingsstrategie

2.1 Invoerfiles:

Het programma 'over' heeft 2 invoerfiles nodig om te kunnen starten, te weten 'options', waarin de opties die besproken zijn in paragraaf 1.2 vastgelegd zijn en een invoerfile waarin de te overhoren woorden met hun vertaling staan. De naam van deze laatste file moet ook in 'options' worden meegegeven.

2.2 Schermindeling:

Nadat bovenstaande files ingelezen c.q. verwerkt zijn, maakt het programma een frame met twee subwindows op het beeldscherm waarin random de te overhoren woorden geplaatst worden. Bovendien zijn er nog 2 windows in het frame, links en rechtsonder. Linksonder is een scherm geschikt voor het invoeren van de achternaam van de proefpersoon en voor eventuele foutmeldingen. Rechtsonder komen, afhankelijk van de opties, een aantal buttons te staan om het programma te starten of te stoppen. Eventueel komt er een button om het volgende te overhoren woord aan te vragen en ook optioneel een button om het aantal woorden aan te geven dat reeds overhoord is.

2.3 Schermvulling:

De te overhoren woorden uit de invoerfile worden apart ingelezen. De woorden staan als volgt in de invoerfile:

```
woord;vertaling
```

Eerst worden alle linkerwoorden, dit zijn dus steeds de woorden tot aan de ';', random ingelezen en vervolgens in het linkerscherm gezet. Tevens wordt hierbij random een positie op het rechterscherm bepaald voor de bijbehorende vertaling. Bij het random bepalen van sommige zaken moet steeds worden bijgehouden of een betreffend woord al ingelezen is en of een zekere positie al bezet is.

2.4 Overhooralgoritmen:

Nadat de proefpersoon de startbutton heeft aangeklikt en

zijn achternaam heeft ingevoerd begint het programma met het overhoren van de woorden, totdat de stopbutton is aangeklikt, of het maximaal aantal te overhoren woorden bereikt is, of het algoritme afgelopen is.
We hebben 4 verschillende algoritmen gemaakt, waarin de volgorde van de aangeboden woorden bepaald wordt. Bij de opties moet een keuze gemaakt worden uit deze algoritmen:

Algoritme 1: overhoort constant alle woorden, onafhankelijk van het antwoord.
Algoritme 2: overhoort de woorden totdat de proefpersoon ze 1 maal goed heeft beantwoord.
Algoritme 3: dit is de vrije optie. De proefpersoon overhoort zelf de woorden, d.w.z. hij of zij klikt zowel in het linker- als in het rechterscherm een woord aan.
Algoritme 4: houdt bij of een antwoord goed of fout is. Iedere keer als u een goed antwoord heeft gegeven wordt de teller bij dat woord een opgehoogd. Heeft u dat antwoord fout dan wordt de teller bij dat woord op nul gezet. Dit algoritme overhoort nu steeds een van de woorden met de laagste tellerstand.

2.5 De uitvoerfile:

Nadat de proefpersoon zijn of haar achternaam ingevoerd heeft, wordt deze naam samen met de vastgelegde opties naar een uitvoerfile geschreven. Vervolgens wordt de datum, dag en tijd in deze file geschreven. De uitvoerfile krijgt als naam:

<achternaam van proefpersoon><nummer>

De achternaam wordt door de persoon zelf ingegeven en het nummer wordt door het programma toegekend. Dit nummer zal tussen de 1 en de 9 liggen naar gelang het aantal uitvoerfiles in de directorie die beginen met dezelfde achternaam. Vervolgens wordt ieder tijdsstip, in seconden vanaf de start, dat een nieuw woord aangeboden wordt, en ieder tijdsstip, in seconden vanaf de start, dat een antwoord wordt aangeklikt, naar deze file geschreven.
Achter ieder weggeschreven tijdsstip van het antwoord komt het regelnummer van de invoerfile te staan van het woord dat gevraagd werd en het regelnummer van de invoerfile van het antwoord dat gegeven werd. Vervolgens komt er achter te staan of het antwoord goed dan wel fout is. Dit wordt aangegeven met een '1' respectievelijk een '0'.

2.6 De opties:

Voordat het overhoorprogramma opgestart kan worden moeten een aantal opties worden vastgelegd. De meeste van deze opties hebben betrekking op de opmaak van het beeldscherm. Dit scherm komt er als volgt uit te zien:

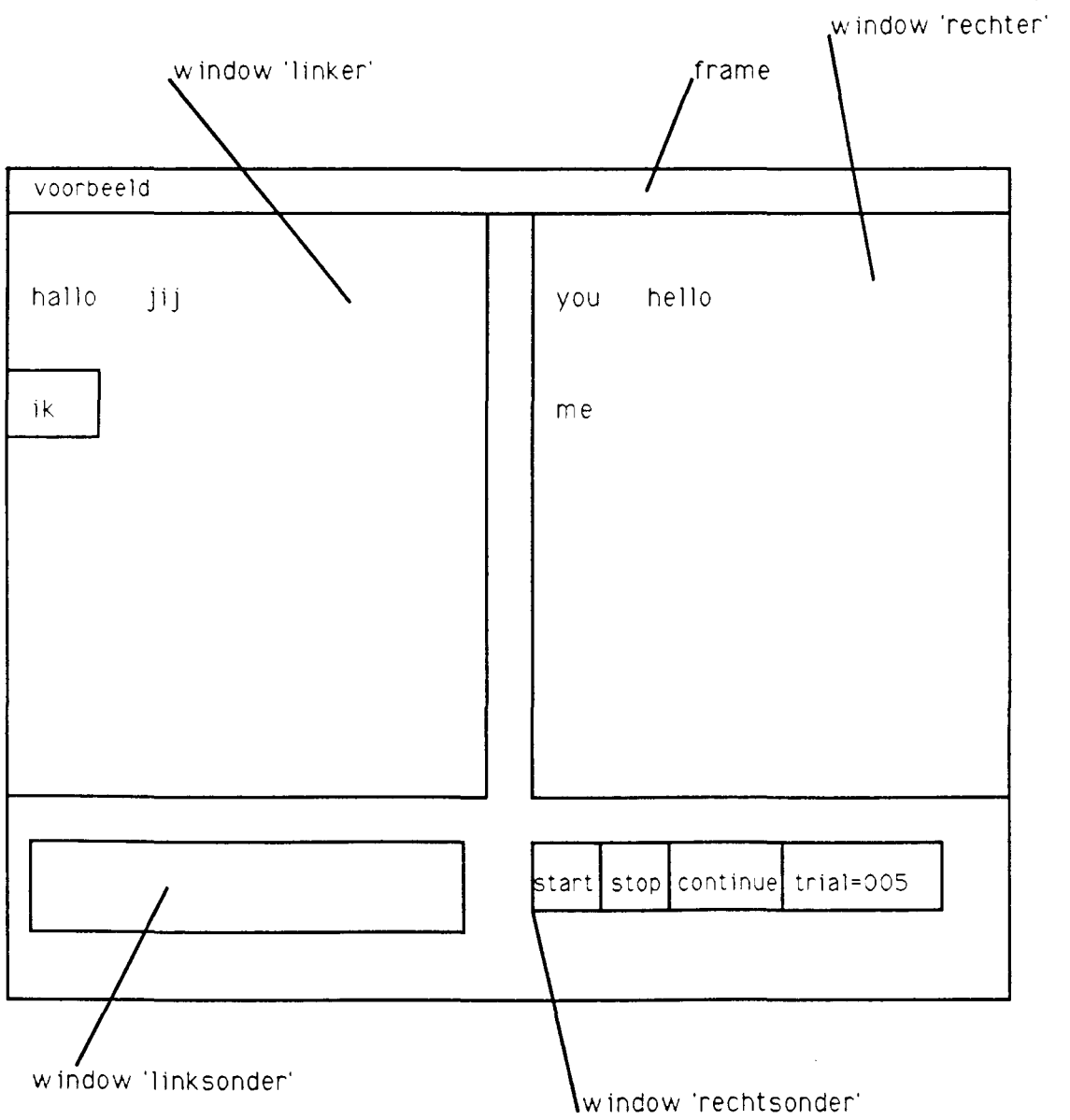

fig. 1: Schermindeling.

Het aantal rijen en kolommen in de figuur kan worden vastgelegd. Bovendien kan worden vastgelegd of u wel of niet wilt dat het programma het goede antwoord aanduidt.
Voorts zijn de buttons 'continue', 'stop' en 'trial' optioneel.

Hfdst. 3: Implementatie

In dit hoofdstuk worden allereerst alle globale variabelen besproken welke in het programmma gebruikt zijn. Vervolgens wordt in de navolgende paragrafen alle procedures beschreven in hun functionele volgorde.

3.1 Globale variabelen:

dispansw-geeft aan of we wel of niet willen dat het programma het goede antwoord geeft.
inpfilename[20]-hierin staat de naam van de invoer file met de te overhoren woorden.
uitv[16]-bevat de naam van de uitvoerfile.
contbut-geeft aan of we wel of niet met de continue-button willen werken.
trialbut-geeft aan of we wel of niet een teller in beeld willen hebben, die aangeeft met welke trial we bezig zijn.
stopbut-geeft aan of we de mogelijkheid geven voortijdig met het programma te stoppen.
poging[10]-bevat de string die aangeeft met het hoeveelste woord we bezig zijn te overhoren. Poging[10] is bedoeld voor uitvoer naar het beeldscherm.
trial-geeft aan met de hoeveelste trial we bezig zijn.
maxtrial-geeft aan hoeveel woorden we maximaal overhoren.
nrrows-geeft aan hoeveel rijen met vreemde woorden we op het beeldscherm willen.
nrcols-geeft aan hoeveel kolommen met vreemde woorden we op het beeldscherm willen.
whichalg-geeft aan met welk algoritme van aanbieding we werken.
maxline-geeft aan uit hoeveel regels de invoer file met de vreemde woorden bestaat.
maxi-door de procedure Radom wordt een willekeurig getal gekozen, modulo maxi.
allreadyline[65]-geeft aan of deze regel reeds uitgekozen is.
allreadyplace[65]-geeft aan of deze plaats op het beeldscherm reeds gevuld is.
selectie-bevat de positie van het aangeklikte woord.
allreadypermunr_*-geeft aan dat deze waarde reeds door Radom uitgekozen is.

afdrukken[65]-geeft de volgorde van de nieuwe reeks van te overhoren woorden weer.

afdrukaantal-geeft weer hoeveel woorden van het array 'afdrukken' overhoord moet worden.

ant[65][3]-in 'ant[i][1]' staat voor elke positie(i) op het linker beeldscherm de bijbehorende plaats op het rechter beeldscherm van het goede antwoord. In 'ant[i][2]' staat of u het antwoord goed, danwel fout heeft aangeklikt.

placel[8]-bevat de coordinaten voor een vierkant om een woord op het linker scherm.

placer[8]-bevat de coordinaten voor een vierkant om een woord op het rechter scherm.

te-geeft aan met welke plaats we in het array 'afdrukken' bezig zijn.

aangeklikt-geeft aan of een woord al dan niet is aangeklikt op het rechter scherm nadat er een woord op het linker scherm omlijnd is.

lengtes[65][2]-lengtes[i][0] bevat de lengte van het woord op positie i op het linker scherm. Lengtes[i][1] bevat die van het rechter scherm.

beginmaar-geeft aan of de achternaam van de proefpersoon al of niet ingevoerd en verwerkt is en het programma begonnen is.

k[65][2]-op de plaats k[t][0] staat het regelnummer welke verwijst naar de inputfile. Het rechterwoord in deze regel komt op positie t in het rechter scherm te staan. K[t][1] bevat een 1 of een 0 om aan te geven of dat woord al of niet meer dan 1 keer voorkomt in dit array.

w[65]-w[t] bevat het regelnummer van de invoerfile welke het rechter woord bevat dat op positie t op het rechter scherm komt te staan. Het array bevat de daadwerkelijk afgedrukte woorden.

l[65]-l[t] bevat het regelnummer van de invoerfile welke het rechter woord bevat dat op positie t op het rechter scherm komt te staan. Het array bevat de woorden zoals die random bepaald zijn. Het array kan dus dubbele woorden bevatten.

placew[8]-bevat de coordinaten voor een ruit op het beeldscherm om een woord.

stop-geeft aan of we de stop-button wel of niet aangeklikt hebben.

aantalwoorden-bevat het tot dan toe aantal daadwerkelijk afgedrukte woorden op het rechter scherm.

secdel-geeft aan hoeveel seconden delay we hebben
tussen uw antwoord en het volgende te over-
horen woord.
start-de tijd in seconden vanaf Januari 1970.
frame-dit is het basis-window welke alle subwin-
dows bevat.
R[65]-bevat de item's welke op het rechter scherm
staan.
L[65]-bevat de item's welke op het linker scherm
staan.
warning,warning2-item's welke foutmeldingen bevatten.
naam_item-item welke de achternaam van de proefper-
soon zal bevatten.
teller-item welke de teller op het beeldscherm
bevat.
linker-subwindow.
rechter-subwindow.
linksonder-subwindow.
rechtsonder-subwindow.
*screen-bevat informatie die nodig is om op een
file te tekenen.

3.2 Opstarten:

Begin(item,event)

Dit is de notify-procedure van de panel-button 'start'. Hierin wordt 'ant[t][2]' geinitialiseerd en de procedure tekstscherm aangeroepen. Deze maakt een subwindow linksonder op het beeldscherm. Hierin kan dan de naam van de gebruiker worden ingevoerd. Indien al eerder 'start' is aangeklikt, en het programma dus al loopt, wordt er een foutmelding gegenereerd. Nadat 'start' dus al eens is aangeklikt wordt de globale variabele 'stop' op nul gezet.

naam_proc(item,event)

De gebeurtenissen in deze procedure kunnen globaal in drie stukken worden verdeeld.
In het eerste gedeelte, dat is het gedeelte tot en met de regel met 'closedir(dirp)', wordt de naam van de uitvoerfile bepaald zoals beschreven is in paragraaf 2.5. Deze naam wordt toegekend aan de globale variabele 'uitv'. Nadat dit gebeurd is wordt het scherm linksonder vernietigd en de procedure messagescherm() aangeroepen waarin een nieuw scherm linksonder wordt gemaakt voor eventuele foutmeldin-

gen. Vervolgens wordt de globale variabele 'beginnen' op 1 gezet ten einde aan te geven dat de achternaam van de proefpersoon is ingevoerd en verwerkt.
In het tweede gedeelte wordt nu de betreffende uitvoerfile geopend en worden enkele gegevens weggeschreven, zoals de achternaam van de proefpersoon en alle parameters. Tevens wordt de globale variabele 'start' bepaald, de starttijd, en weggeschreven.
In het derde deel, de laatste regel van de procedure wordt Initialisatie() aangeroepen. Hiermee start het eigenlijke programma.

tekstscherm()

Hierin wordt een scherm linksonder gegenereerd. Hierin wordt een panel-item van het type PANEL_TEXT gezet om het mogelijk te maken om de achternaam in te voeren. Tevens wordt de cursor op het beeldscherm op dit panel-item gezet zodat de naam meteen ingevoerd kan worden.

3.3 Inlezen:

Positioneer(regelnummer)

Deze procedure zet de inputstream aan het begin van de regel, die door Controlline() bepaalt is, neer in de invoerfile. Deze regel wordt aangegeven met 'regelnummer'. Deze procedure heeft verder geen invloed op globale variabelen.

Readoptions()

In deze procedure worden alle opties ingelezen, die in het programma opties zijn vastgelegd. De volgende variabelen worden ingelezen:

'nrrows'-geeft aan hoeveel rijen met vreemde woorden u op het beeldscherm wilt.
'nrcols'-geeft aan hoeveel kolommen met vreemde woorden u op het beeldscherm wilt.
'dispansw'-geeft aan of u wel of niet wilt dat het programma het goede antwoord geeft.
'whichalg'-geeft aan welk algoritme u gekozen heeft voor de aanbieding van

de te overhoren woorden.
'contbut'-geeft aan of u wel of niet met de continue button wilt werken. Wanneer u niet met deze button werkt heeft de variabele secdel een waarde.
'secdel'-geeft aan hoeveel seconden delay u wilt tussen uw antwoord en het volgende woord.
'maxtrial'-geeft aan hoeveel woorden overhoord mogen worden.
'trialbut'-geeft aan of u wel of niet wilt dat het aantal woorden dat overhoord is op het beeldscherm staat.
'stopbut'-geeft aan of u wel of niet de mogelijkheid geeft om het programma te stoppen voordat maxtrial bereikt is.
'inpfilename'-hierin staat de naam van de invoerfile.

Readinputfile()

In deze procedure wordt bepaalt uit hoeveel regels de invoerfile bestaat. Dit aantal wordt aan 'maxline' toegekend.

3.4 Schermvulling:

MaakItemL(t,x,y,woord)

Deze procedure creeert een panel-button op het linker scherm. De inhoud van de panel-button is de meegegeven variabele 'woord'. De panel-button wordt neergezet op de karaktercoordinaat (x,y). Tevens wordt in deze procedure de gecreeerde panel-button onthouden in het array L op plaats t voor eventuele latere identificatie van de buttons.

MaakItemR(t,x,y,woord)

Voert hetzelfde uit als MaakItemL(t,x,y,woord) met het verschil dat de panel-button nu op het rechterscherm wordt geplaatst en onthouden wordt in array R.

Hoofdprog()

Deze procedure roept twee keer de procedure Hulpprog aan voor het vullen van respectievelijk het linker en het rechter scherm. Tevens wordt in deze procedure van te voren de array's allreadyline en allreadyplace geinitialiseerd, welke worden gebruikt na de aanroep van Hulpprog(0).

Hulpprog(zijde)

In deze procedure wordt voor zijde==0 het linker scherm en voor zijde==1 het rechter scherm gevuld. In deze specifieke toepassing van Hulpprog heeft deze procedure aleen zin voor het vullen van het hele beeldscherm als dan ook eerst Hulpprog(0) en daarna Hulpprog(1) wordt aangeroepen, omdat in Hulpprog(0) enkele array's gevuld worden, die Hulpprog(1) gebruikt.

De belangrijkste lokale variabelen van de procedure zijn:

(int)maxlengtewoord: Deze variabele bevat de maximale woordlengte van de woorden welke in een kolom staan. Dit is nodig voor het bepalen van de plaats op het beeldscherm van de volgende kolom met woorden.

(int) a,b: Dit zijn de coordinaten (x,y) in een subwindow waar telkens een panel-button , lees woord, wordt geplaatst.

(int)regelnummer: Bevat het random gekozen regelnummer van de inputfile van waaruit het linker woord wordt ingelezen en geplaatst wordt op het linker scherm.

(int)positie: Bevat de random gekozen positie op het rechter scherm van het rechter woord, in de regel van de invoerfile aangewezen door 'regelnummer'.

(int)hulp: Deze variabele kan de waarde 1 of -1 aannemen. De waarde 1 geeft aan dat er op het rechter scherm een button c.q. woord wordt neergezet nadat gecontrolleerd is dat dat woord nog niet eerder voorkwam op het rechter scherm.
De waarde -1 geeft aan dat het woord wel eerder voorkwam en dat er daarom geen nieuwe panel-button met dat woord wordt gemaakt.

(int)aantal_buttons: Geeft het aantal buttons weer dat, tot op dat moment, gemaakt is op het rechter scherm.

De procedure Hulpprog kan de globale variabelen (array's) l, k, ant, w en lengtes veranderen, in die zin dat ze gevuld worden met de daar toe bestemde informatie.

De globale werking van Hulpprog, gesplitst voor zijde==0 en zijde==1, is als volgt:

zijde==0:

Er wordt random een regelnummer bepaald en random een positie op het rechter scherm. Uit de regel van de invoerfile wordt het linker woord gelezen en daar wordt een button van gemaakt welke geplaats wordt op positie t.

zijde==1;

De rechter woorden zijn al toebedeeld aan de posities op het rechter scherm volgens 'k[t][0]'. Ze worden nu slechts gelezen uit de invoerfile. Vervolgens wordt bepaald of ze wel of niet al eens geplaatst zijn op het rechter scherm d.m.v. de aanroep van procedure Dubbelrechts. Als dat niet zo is (hulp==1) dan wordt er een panel-button gecreeerd, anders niet.

Radom(y)

Deze procedure kiest een random getal tussen 0 en maxi en geeft deze als uitvoer mee aan y.
Om steeds bij een nieuwe opstart van het programmma een nieuwe reeks random getallen te genereren wordt het seed van de standaard procedure rand verandert. Dit gebeurt door de tijd van de dag aan srand mee te geven.

Controlline(regelnr)

Deze procedure bepaalt welke regel uit de invoerfile ingelezen wordt. Radom wordt aangeroepen om een regel (line) te bepalen die ingelezen kan worden. Omdat Radom een getal bepaalt modulo 'maxi' wordt eerst aan 'maxi' het aantal regels toegekend (plus 1) waaruit de invoerfile bestaat. Wanneer in het array 'allreadyline[line]' een 1 staat op de plaats van de door Radom bepaalde line, dan is deze regel reeds ingelezen.
Er moet dus een nieuwe line bepaalt worden door Radom opnieuw aan te roepen. Op deze plaats in het array 'allreadyline[line]' wordt dan een 1 gezet. De bepaalde line wordt aan 'regelnr' toegekend en als uitvoer van de proce-

dure meegegeven.

Controlplace(positie)

Deze procedure werkt exact hetzelfde als de procedure Controlline, met dien verstande dat Controlplace bijhoudt welke posities op het rechter beeldscherm reeds gevuld zijn. En als uitvoer aan 'positie' een vrije plaats op het rechter beeldscherm meegeeft.

Dubbel_rechts(t,woord1)

Deze procedure controleert of een woord, dat meegegeven is in de variabele 'woord1', op het rechter scherm tot nu toe al eens is afgedrukt.
Dit gebeurt door het array 'k' af te lopen tot de plaats t, de plaats van het huidige woord. Indien een bepaald woord al eens vaker is voorgekomen staat er achter dat woord in kolom twee van 'k' een 1. BIj de controle in array 'k' naar het eerder voorgekomen zijn van 'woord1' hoeft dus 'woord1' niet vergeleken te worden met woorden waarna door 'k' verwezen wordt in de invoerfile waarachter een 1 staat. Indien blijkt dat een woord al eens is afgedrukt wordt het array 'ant' aangepast. De waarde van ant[j][1] die wees naar de panel-button die dus nu niet wordt afgedrukt moet nu verwijzen naar een andere, eerder voorkomende panel-button. Indien 'woord1' dubbel voorkomt wordt de waarde -1 geretourneerd, indien niet dan de waarde 1.

3.5 Overhooralgoritmen:

Selecteer(sel,item,zijde)

Deze procedure bepaalt bij aanroep, welk item is aangeklikt. Hiervoor wordt het item-kenmerk meegegeven in 'item'. Bovendien moet worden meegegeven over welke zijde op het beeldscherm het gaat (zijde==0 voor linker scherm en zijde==1 voor rechter scherm). Voor de bepaling van de positie wordt gebruik gemaakt van het array 'L' of het array 'R', welke gevuld werden in de procedure Hulpprog. Deze positie voor het item wordt opgeslagen in de variabele 'sel'.

Alg_1()

Deze procedure bepaalt de volgorde van alle te overhoren woorden,door de procedure Radom aan te roepen totdat alle plaatsen op het beeldscherm gevuld zijn. Om er voor te zorgen dat in iedere reeks van te overhoren woorden elk

woord maar 1 maal voorkomt, wordt het array 'allready-permunr_1' op 1 gezet als het desbetreffende woord geselecteerd is. De bepaalde reeks wordt aan het array 'afdrukken' toegekend.

Alg_2()

In het array 'ant[i][2]' staat of u het bijbehorende woord reeds goed ('ant[i][2]'==1) of fout ('ant[i][2]'==0) beantwoord heeft. Dit algoritme bepaalt een reeks van te overhoren woorden die u nog niet of fout beantwoord heeft. Met behulp van het array 'allreadypermunr_2' wordt ervoor gezorgd dat ieder woord maximaal 1 maal voorkomt in de reeks. De gevonden reeks wordt toegekend aan het array 'afdrukken'.

Alg_4()

Iedere keer als u een goed antwoord geeft wordt de desbetreffende 'ant[i][2]' met 1 opgehoogt,wanneer u een foutief antwoord geeft wordt 'ant[i][2]' op 0 gezet. Deze procedure bepaalt de laagste waarde van het hele array 'ant[i][2]' en kiest hieruit random een woord dat overhoort moet worden. Dit geselecteerde woord wordt aan 'afdrukken[1]' toegekend. Vervolgens wordt een vierkant om dit woord getekend en de tijd weggeschreven waarop dit vierkant geplaatst werd.

Initialisatie()

Nadat een reeks van woorden is afgehandeld moeten de vierkantjes van de laatste woorden weggehaald worden. Vervolgens moeten, alvorens we een nieuwe reeks bepalen, enkele array's opnieuw geinitialiseerd worden. Als deze procedure het bovenstaande heeft uitgevoerd wordt het geselecteerde algoritme aangeroepen. En eventueel (algoritme 1 en 2) een vierkantje, om het te overhoren woord getekend en de daarbij behorende tijd weggeschreven.

messagescherm()

In deze procedure wordt er een scherm linksonder op het beeldscherm geplaatst. Tevens worden er twee panel-items gemaakt, in feite dus twee foutmeldingen, welke echter niet gedisplayed worden.

teller_ophoging()

Deze procedure 'verhoogt' de globale variabele 'trial'. Deze 'trial' is een characterstring. Indien nu in deze variabele stond '003', dan staat er na aanroep van deze procedure in deze variabele '004'. Uiteindelijk wordt in de globale variabele 'poging', ook een characterstring, de letters 'trial=' gezet met daarachter de inhoud van 'trial' in dit geval dus 'trial=004'. Dit woord kan dan aan de trial-button worden toegekend en op die manier op het beeldscherm worden afgedrukt.
Indien trial de waarde '999' overschrijdt wordt de procedure einde() aangeroepen.

Rechter_scherm_proc(item,event)

Dit is de notify-procedure van de panel-buttons op het rechter scherm.
In deze procedure wordt allereerst bepaald welke button is aangeklikt door de aanroep van Selecteer. Vervolgens wordt 'ant' bijgewerkt en de trial-button opgehoogd. Daarna wordt er een vierkant en een ruit geplaatst op het rechter scherm. De globale variabele 'aangeklikt' wordt op 1 gezet ten einde aan te geven dat er een rechter woord is aangeklikt.
Vervolgens worden de regelnummers weggeschreven waarin in de inputfile het gevraagde woord en het antwoord zijn terug te vinden. Tevens wordt weggeschreven of het antwoord goed of fout was. Aan het begin van de procedure was de tijd al weggeschreven.
Vervolgens wordt gekeken of er al dan niet een continue-button aanwezig is en u al of niet met algoritme 3 bezig bent. Indien dit namelijk niet zo is wordt er een tijdsdelay gemaakt voordat de procedure voortgang wordt aangeroepen. In de procedure wordt verder nog getest of 'maxtrial' bereikt is. Indien dit het geval is wordt de procedure einde() aangeroepen.

Voortgang(item, event)

Deze procedure wordt aangeroepen zodra de continue-button wordt aangeklikt of de maximale delay tijd verstreken is. Eerst moet een eventuele warning die kwam door het voorgaande antwoord, weggehaald worden. Ook moet de trialbutton met 1 opgehoogd worden. Vervolgens moet het weer mogelijk gemaakt worden dat er een nieuw antwoord gegeven wordt en de oude vierkantjes en ruit moeten gewist worden. Als we met algoritme 4 werken moet er een nieuw te overhoren woord

bepaald worden, en een vierkantje geplaatst worden, dit gebeurt door Alg_4 aan te roepen. In de andere gevallen moet om het volgende woord, aangegeven door 'afdrukken[nummer]', een vierkantje geplaatst en de bijbehorende tijd weggeschreven worden. Als alle woorden van afdrukken geweest zijn, te>afdrukaantal, dan wordt initialisatie() opnieuw aangeroepen om een nieuwe volgorde van afdrukken te bepalen.

Linker_scherm_proc(item, event)

Deze procedure wordt aangeroepen zodra op het linker scherm een woord aangeklikt wordt. Dit kan alleen in de vrije optie, algoritme 3. Eerst moet dan een eventuele foutmelding ten gevolge van het vorige antwoord weggehaald worden. De trialbutton moet met 1 opgehoogd worden en de oude vierkantjes en ruit moeten verwijdert worden. Vervolgens moet het onmogelijk gemaakt worden om nog een woord op het linker scherm aan te klikken. Maar wel op het rechter scherm. Tenslotte moet er om het hier geselecteerde woord een vierkantje geplaatst en de bijbehorende tijd weggeschreven worden.

3.6 Schermbewerking:

vierkant(plaats)

Deze procedure plaatst een vierhoek van zwarte lijnen op het beeldscherm met de coordinaten, in pixels, welke gegeven zijn door het array 'plaats' volgens:

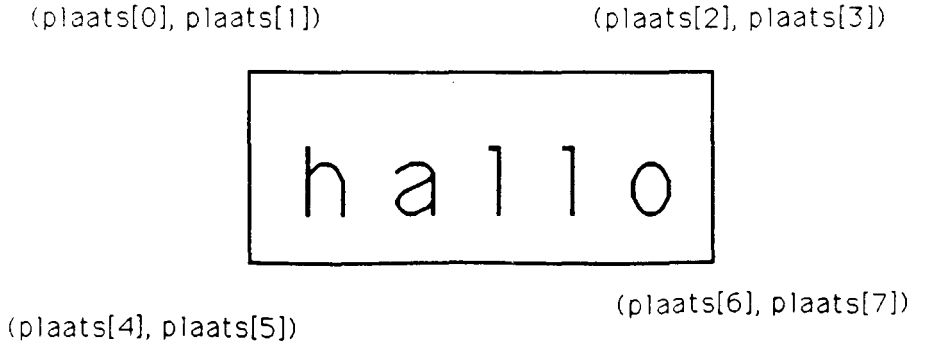

fig. 2: Vierhoek om woord op beeldscherm.

wisvierkant(plaats)

Bij aanroep plaatst deze procedure een vierhoek van witte lijnen op het beeldscherm met behulp van het array 'plaats' volgens figuur 2. De opzet van deze procedure is, om bij aanroep een vierkant van witte lijnen te plaatsen over een vierkant van zwarte lijnen, hiermee de indruk wekkende dat dit laatste vierkant gewist geworden is.

plaatsvierkant(zijde)

Deze procedure bepaalt de 4 (x,y)-coordinaten van een vierkant op het beeldscherm om een bepaald panel-item c.q. woord, volgens figuur 2. Dit gebeurt voor een woord op het linker of rechter scherm voor 'zijde'==0, respectievelijk 'zijde'==1.
Voor het linker scherm gebeurt dit aan de hand van het woord dat omlijnd moet worden volgens de inhoud van 'afdrukken[nummer]'. Voor het rechter scherm gebeurt dit volgens het aangeklikte woord op dit scherm, de waarde van 'selectie'.

plaatsruit()

Bij deze procedure worden de coordinaten van een ruit bepaald op het rechter scherm en zet die in 'placew'. Deze coordinaten worden bepaald voor het woord dat aangeklikt had moeten worden, het goede antwoord. De positie hiervan wordt bepaald door de inhoud van 'ant[afdrukken[te]][1]]'. De ruit met zijn coordinaten hebben dan onderstaande gedaante:

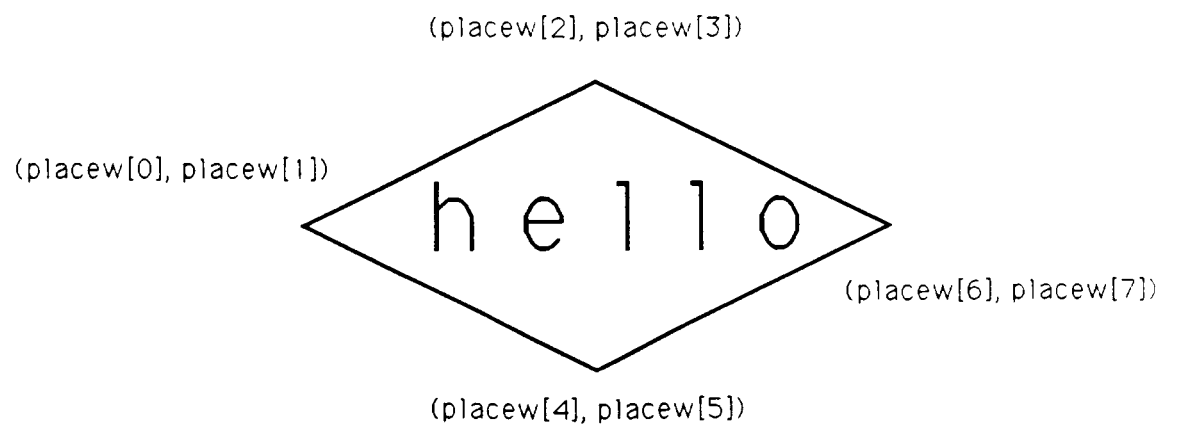

fig. 3: Ruit om woord op beeldscherm.

3.7 Beeindiging:

Einde(item,event)

Dit is de notify-procedure van de stop-button. Als deze wordt aangeklikt, wordt de globale variabele 'stop' op 1 gezet om het mogelijk te maken start opnieuw aan te klikken, als 'beginmaar' ongelijk is aan 1, d.w.z. dat er nog geen naam was ingevoerd. Indien 'beginmaar'==1 en dus de naam wel is ingevoerd en het programma is begonnen, dan wordt bij het aanroepen van de procedure Einde de outputfile geclosed en het frame vernietigd.

3.8 Uitvoerfile:

Tijd()

Deze procedure schrijft zodra hij wordt aangeroepen de tijd van aanroep ten opzichte van de opstart van het programma in seconden naar de uitvoerfile van het programma.

Hfdst. 4: Vastleggen van de opties.

In bijlage 2 vindt u de uitdraai van het programma opties.c. De parameters die u met dit programma kunt instellen zijn in het voorgaande reeds verklaard. U kunt dit nog eens nalezen op page 1 van de uitdraai op bijlage 2. Als u het programma opties opstart, vraagt het programma u eerst of u uitleg wilt over de in te stellen parameters. Vervolgens moet u de instelling invoeren, waarbij steeds getest wordt of de door u gewenste instelling mogelijk is. Is deze instelling niet mogelijk, dan wordt door het programma een andere instelling gevraagd. De volgende randvoorwaarden moeten voor de instelling gelden:

0<number of rows<17
0<number of colums<5
0<which algorithem<5
0<seconds delay<20
000<max number of trials<=999

De uitvoer van opties, de parameter instelling, wordt naar de file "options" geschreven.

Hfdst. 5: Gebruikershandleiding:

5.1 Doel van het programma:

Het doel van het programma 'over' is het testen met welk algoritme van aanbieding van gepaarde associatieven, deze het meest effectief aangeleerd kunnen worden.

5.2 Opstarten:

Het programma over is geschikt voor een SUN 3/50 terminal. Wanneer u in de /usr/rooy directory bent zijn de programma's over en opties voor u beschikbaar. Deze programma's kunt u runnen met de volgende commando's:

```
helena# opties
helena# over
```

5.3 Invoer:

In het programma opties moet u zoals reeds vermeld enkele parameter instellingen geven, waarover het programma u zelf uitleg geeft. Het programma over vraagt u enkel om de <start> button, rechtsonder, met de muis aan te klikken en vervolgens, linksonder, uw achternaam in te voeren. Daarna begint over met het overhoren van de gepaarde associatieven op de manier zoals in het programma opties is vastgelegd. Het programma over gebruikt nog een invoerfile, waar de te overhoren woorden in staan. Deze file moet op de volgende manier gevuld zijn:

```
woord;vertaling
```

De naam van deze file moet ook bij de opties meegegeven worden.

5.4 Uitvoer:

De uitvoer van het programma over staat in de file :

```
<achternaam><nummer>
```

Hierin staan alle opties en datum en tijd van het opstarten van het programma. En verdere gegevens over de resultaten van de proefpersoon zoals die beschreven zijn in paragraaf 2.5.

Bijlage 1.


```
#include <suntool/sunview.h>
#include <suntool/panel.h>
#include <stdio.h>
#include <sys/time.h>
#include <sys/types.h>
#include <sys/dir.h>
#include <string.h>

static char dispansw,inpfilename[20],uitv[16],contbut,trialbut,stopbut;
static char poging[10],trial[3],maxtrial[3];
static int nrrows,nrcols,whichalg,maxline,maxi;
static int allreadyline[65],allreadyplace[65],selectie;
static int allreadypermunr_1[65],allreadypermunr_2[65],allreadypermunr_4[65];
static int afdrukken[65],afdrukaantal,ant[65][3];
static int placel[8],placer[8],te,aangeklikt;
static int lengtes[65][2],beginmaar,k[65][2],w[65],l[65];
static int placew[8],stop,aantalwoorden;
static long secdel;
static double start;
static Frame frame;
static Panel_item R[65],L[65],warning,warning2,naam_item,teller;
static Panel linker,rechter,linksonder,rechtsonder;
struct pixrect *screen;
FILE *fopen(), *fp;

static void Hoofdprog();
static void Hulpprog();
static void Radom();
static void Controlline();
static void Controlplace();
static void Selecteer();
static void Positioneer();
static void Readoptions();
static void Readinputfile();
static void MaakItemL();
static void MaakItemR();
static void Alg_1();
static void Alg_2();
static void Alg_4();
static void Initialisatie();
static void Linker_scherm_proc();
static void Rechter_scherm_proc();
static void voortgang();
static void vierkant();
static void plaatsvierkant();
static void wisvierkant();
static void plaatsruit();
static void begin();
static void einde();
static void naam_proc();
static void messagescherm();
static void tekstscherm();
static void tijd();
static void teller_ophoging();
int Dubbel_rechts();

main(argc, argv)
int argc;
char **argv;
{
 int i;
 strcpy(trial,"001");
 strcpy(poging,"trial=");
 strcat(poging,trial);
```

```
 Readoptions();
 Readinputfile();
 frame=window_create(0,FRAME,FRAME_LABEL,"voorbeeld",
                     WIN_X,0,
                     WIN_Y,0,
                     WIN_HEIGHT,880,
                     WIN_WIDTH,1090,
                     0);
 linker=window_create(frame,PANEL,
                      WIN_X,0,
                      WIN_Y,0,
                      WIN_WIDTH,545,
                      WIN_HEIGHT,780,
                      0);
 rechter=window_create(frame,PANEL,
                       WIN_X,550,
                       WIN_RIGHT_OF,linker,
                       WIN_HEIGHT,780,
                       0);
 rechtsonder=window_create(frame,PANEL,
                           WIN_X,550,
                           WIN_Y,800,
                           WIN_HEIGHT,70,
                           0);
 panel_create_item(rechtsonder,PANEL_BUTTON,
                   PANEL_LABEL_IMAGE,panel_button_image(rechtsonder,"start",0,0),
                   PANEL_NOTIFY_PROC,begin,
                   0);
 if (stopbut=='y')
     panel_create_item(rechtsonder,PANEL_BUTTON,
                       PANEL_LABEL_IMAGE,panel_button_image(rechtsonder,"stop",0,0),
                       PANEL_NOTIFY_PROC,einde,
                       0);
 window_set(linker,WIN_CONSUME_PICK_EVENT,WIN_NO_EVENTS,0);
 window_set(rechter,WIN_CONSUME_PICK_EVENT,WIN_NO_EVENTS,0);
 if ((whichalg!=3) && (contbut=='y'))
     panel_create_item(rechtsonder,PANEL_BUTTON,
                       PANEL_LABEL_IMAGE,panel_button_image(rechtsonder,"continue",0,0),
                       PANEL_NOTIFY_PROC,voortgang,
                       0);
 if (trialbut=='y')
     teller=panel_create_item(rechtsonder,PANEL_BUTTON,
                              PANEL_LABEL_IMAGE,panel_button_image(rechtsonder,poging,0,(
                              0);
 window_fit(rechtsonder);
 aangeklikt=0;
 stop=1;
 for(i=1;i<31;i++)
     k[i][1];
 for(i=0;i<8;i++)
     placel[i]=placer[i]=placew[i]=12;
 Hoofdprog();
 window_main_loop(frame);
}

static void
Radom(y)
int  *y;
{
 long x;
 struct timeval tp;
 struct timezone tzp;
 gettimeofday(&tp, &tzp);
 srand(tp.tv_usec);
```

```
 *y=0;
 while (*y==0)
       {
        x=rand();
        *y=x % maxi;
       }
}

static void
Controlline(regelnr)
int  *regelnr;
{
 int line;
 maxi=aantalwoorden+1;
 do
   Radom(&line);
 while (allreadyline[line]==1);
 allreadyline[line]=1;
 *regelnr=line;
}

static void
Controlplace(positie)
int  *positie;
{
 int place;
 maxi=aantalwoorden+1;
 do
   Radom(&place);
 while (allreadyplace[place]==1);
 allreadyplace[place]=1;
 *positie=place;
}

static void
Positioneer(regelnummer)
int  regelnummer;
{
 int i;
 char woord[62];
 fseek(fp,0L,0);
 for(i=1;i<regelnummer;i++)
     fscanf(fp,"%s",woord);
 fscanf(fp,"\n");
}

static void
Readoptions()
{
 fp=fopen("options","r");
 fscanf(fp,"%d",&nrrows);
 fscanf(fp,"%d",&nrcols);
 fscanf(fp,"%s",&dispansw);
 fscanf(fp,"%d",&whichalg);
 fscanf(fp,"%s",&contbut);
 if (contbut=='n')
     fscanf(fp,"%d",&secdel);
 fscanf(fp,"%s",maxtrial);
 fscanf(fp,"%s",&trialbut);
 fscanf(fp,"%s",&stopbut);
 fscanf(fp,"%s",inpfilename);
 fclose(fp);
}
```

```
static void
Readinputfile()
{
 char zin[80];
 fp=fopen(inpfilename,"r");
 maxline=0;
 while (fscanf(fp,"%s",zin)!=EOF)
        maxline++;
 fclose(fp);
}

static void
MaakItemL(t, x, y, woord)
int x,y,t;
char woord[15];
{
 L[t]=panel_create_item(linker,PANEL_BUTTON,
                        PANEL_LABEL_STRING,woord,
                        PANEL_ITEM_X,ATTR_COL(x),
                        PANEL_ITEM_Y,ATTR_COL(y),
                        PANEL_NOTIFY_PROC,Linker_scherm_proc,
                        0);
}

static void
MaakItemR(t, x, y, woord)
int x,y,t;
char woord[15];
{
 R[t]=panel_create_item(rechter,PANEL_BUTTON,
                        PANEL_LABEL_STRING,woord,
                        PANEL_ITEM_X,ATTR_COL(x),
                        PANEL_ITEM_Y,ATTR_COL(y),
                        PANEL_NOTIFY_PROC,Rechter_scherm_proc,
                        0);
}

static void
Hoofdprog()
{
 int i;
 aantalwoorden=nrrows*nrcols;
 if (maxline<aantalwoorden)
     aantalwoorden=maxline;
 for(i=0;i<=aantalwoorden;i++)
     allreadyline[i]=allreadyplace[i]=0;
 Hulpprog(0);
 Hulpprog(1);
}

static void
Hulpprog(zijde)
int zijde;
{
 int maxlengtewoord,t,a,b,c,regelnummer,positie,rij,i,hulp;
 int aantal_buttons,j;
 char woord[35];
 fp=fopen(inpfilename,"r");
 t=1;
 hulp=1;
 maxlengtewoord=0;
 aantal_buttons=0;
 a=2;
 b=2;
```

```
 c=1;
 rij=1;
 while (t<=aantalwoorden)
   {
    if (zijde==0)
      {
       Controlline(&regelnummer);
       Controlplace(&positie);
       l[t]=regelnummer;
       k[positie][0]=regelnummer;
       ant[t][1]=positie;
       Positioneer(regelnummer);
       for(i=0;(woord[i]=getc(fp))!=';';i++);
       woord[i]='\0';
      }
    else
        {
         Positioneer(k[c][0]);
         c++;
         for(i=0;getc(fp)!=';';i++);
         for(i=0;(woord[i]=getc(fp))!='\n';i++);
         woord[i]='\0';
        }
if(i>maxlengtewoord)
   maxlengtewoord=i;
 if (zijde==0)
    {
     MaakItemL(t,a,b,woord);
     lengtes[t][0]=i;
    }
 else
    {
     if ((hulp=Dubbel_rechts(t,woord))==1)
          {
           aantal_buttons++;
           w[aantal_buttons]=k[t][0];
           MaakItemR(aantal_buttons,a,b,woord);
           lengtes[aantal_buttons][1]=i;
           for(j=1;j<=aantalwoorden;j++)
               {
                if (ant[j][1]==t)
                    ant[j][1]=aantal_buttons;
               }
          }
    }
 if (hulp==1)
     {
      if (rij<nrrows)
          {
           b=b+2;
           rij++;
          }
      else
          {
           b=2;
           a=a+3+maxlengtewoord;
           maxlengtewoord=0;
           rij=1;
          }
     }
 t++;
   }
 fclose(fp);
}
```

```
static void
Alg_1()
{
 int permutatie[65],permunr,i;

 afdrukaantal=aantalwoorden;
 for(i=1;i<=aantalwoorden;i++)
    {
     maxi=aantalwoorden+1;
     do
       Radom(&permunr);
     while (allreadypermunr_1[permunr]==1);
     allreadypermunr_1[permunr]=1;
     afdrukken[i]=permunr;
    }
}

static void
Alg_2()
{
 int permutatie[65],permunr,i;
 int overhooraantal,opslag[65],k;

 overhooraantal=0;
 k=1;
 for (i=1;i<=aantalwoorden;i++)
      opslag[i]=0;
 for (i=1;i<=aantalwoorden;i++)
    {
     if (ant[i][2]==0)
       {
        overhooraantal++;
        opslag[k]=i;
        k++;
       }
     }
 afdrukaantal=overhooraantal;
 for(i=1;i<=overhooraantal;i++)
    {
     maxi=overhooraantal+1;
     do
       Radom(&permunr);
     while (allreadypermunr_2[permunr]==1);
     allreadypermunr_2[permunr]=1;
     afdrukken[i]=opslag[permunr];
    }
}

static void
Alg_4()
{
 int permutatie[65],permunr,i;
 int overhooraantal,opslag[65],k;
 int laagste,gevonden;

 laagste=0;
 gevonden=0;

 do
   {
    for (i=1;i<=aantalwoorden;i++)
       {
        if (ant[i][2]==laagste)
```

```
            gevonden=1;
        }
    laagste++;
   }
 while (gevonden==0);

 overhooraantal=0;
 k=1;
 for (i=1;i<=aantalwoorden;i++)
      opslag[i]=0;
 for (i=1;i<=aantalwoorden;i++)
    {
     if (ant[i][2]==(laagste-1))
       {
        overhooraantal++;
        opslag[k]=i;
        k++;
       }
     }
 afdrukaantal=overhooraantal;
     te=1;
     maxi=overhooraantal+1;
       Radom(&permunr);
     afdrukken[1]=opslag[permunr];
 plaatsvierkant(0);
 vierkant(placel);
 tijd();
}

static void
Initialisatie()
{
 int i;
 wisvierkant(placel);
 wisvierkant(placer);
 wisvierkant(placew);
 for(i=0;i<=maxline;i++)
     allreadypermunr_1[i]=allreadypermunr_2[i]=allreadypermunr_4[i]=0;
 if (whichalg!=3)
     {
      window_set(linker, WIN_CONSUME_PICK_EVENT , WIN_NO_EVENTS, 0);
      if (whichalg==4)
          Alg_4();
      else
         {
          if (whichalg==1)
              Alg_1();
          if (whichalg==2)
              Alg_2();
          if (afdrukaantal==0)
              einde();
          te=1;
          plaatsvierkant(0);
          vierkant(placel);
          tijd();
         }
     }
}

static void
Selecteer(sel,item,zijde)
Panel_item item;
int zijde,*sel;
{
```

```
 int i;
 i=1;
 if (zijde==0)
     {
      while(L[i]!=item)
      i++;
     }
 if (zijde==1)
     {
      while(R[i]!=item)
      i++;
     }
 *sel=i;
}

static void
Linker_scherm_proc(item,event)
Panel_item item;
Event *event;
{
 if (whichalg==3)
     {
      panel_set(warning2,PANEL_SHOW_ITEM,FALSE,0);
      if (trialbut=='y')
           panel_set(teller,PANEL_LABEL_IMAGE,
                     panel_button_image(rechtsonder,poging,0,0),
                     0);
      wisvierkant(placel);
      wisvierkant(placer);
      wisvierkant(placew);
      window_set(linker, WIN_CONSUME_PICK_EVENT , WIN_NO_EVENTS, 0);
      window_set(rechter, WIN_CONSUME_PICK_EVENT , WIN_MOUSE_BUTTONS, 0);
      Selecteer(&selectie,item,0);
      te=1;
      afdrukken[te]=selectie;
      plaatsvierkant(0);
      vierkant(placel);
      tijd();
     }
}

static void
Rechter_scherm_proc(item,event)
Panel_item item;
Event *event;
{
 int antgoed;
 long x;
 struct timeval tp;
 struct timezone tzp;
 tijd();
 panel_set(warning2,PANEL_SHOW_ITEM,FALSE,0);
 panel_set(warning,PANEL_SHOW_ITEM,FALSE,0);
 window_set(rechter, WIN_CONSUME_PICK_EVENT , WIN_NO_EVENTS, 0);
 Selecteer(&selectie,item,1);
 antgoed=0;
 if (selectie==ant[afdrukken[te]][1])
     {
      ant[afdrukken[te]][2]++;
      antgoed=1;
     }
 else
      ant[afdrukken[te]][2]=0;
 teller_ophoging();
```

```
 if (whichalg==3)
     window_set(linker, WIN_CONSUME_PICK_EVENT , WIN_MOUSE_BUTTONS, 0);
 plaatsruit();
 plaatsvierkant(1);
 vierkant(placew);
 vierkant(placer);
 aangeklikt=1;
 fprintf(fp,"%d %d ",l[afdrukken[te]],w[selectie]);
 fprintf(fp,"%d\n",antgoed);
 if (strcmp(trial,maxtrial)>0)
     einde();
 if ((contbut=='n') && (whichalg!=3))
    {
     gettimeofday(&tp,&tzp);
     x=tp.tv_sec;
     do
       gettimeofday(&tp,&tzp);
     while ((tp.tv_sec-x)<=(secdel));
     voortgang();
    }
}

static void
voortgang(item,event)
Panel_item item;
Event *event;
{
 panel_set(warning2,PANEL_SHOW_ITEM,FALSE,0);
 if (beginmaar==1)
     {
      if (aangeklikt==0)
          panel_set(warning,PANEL_SHOW_ITEM,TRUE,0);
      else
          {
           aangeklikt=0;
           if (trialbut=='y')
               panel_set(teller,PANEL_LABEL_IMAGE,
                         panel_button_image(rechtsonder,poging,0,0),
                         0);
           window_set(rechter, WIN_CONSUME_PICK_EVENT , WIN_MOUSE_BUTTONS, 0);
           wisvierkant(placer);
           wisvierkant(placel);
           wisvierkant(placew);
           if (whichalg==4)
               Alg_4();
           else
               {
                te++;
                if (te>afdrukaantal)
                    Initialisatie();
                plaatsvierkant(0);
                vierkant(placel);
                tijd();
               }
          }
     }
}

static void
vierkant(plaats)
int plaats[8];
{
 screen=pr_open("/dev/fb");
 pr_vector(screen,plaats[0],plaats[1],plaats[2],plaats[3],PIX_SRC|PIX_COLOR(1),1);
```

```
 pr_vector(screen,plaats[4],plaats[5],plaats[6],plaats[7],PIX_SRC|PIX_COLOR(1),1);
 pr_vector(screen,plaats[0],plaats[1],plaats[4],plaats[5],PIX_SRC|PIX_COLOR(1),1);
 pr_vector(screen,plaats[2],plaats[3],plaats[6],plaats[7],PIX_SRC|PIX_COLOR(1),1);
 pr_close(screen);
}

static void
wisvierkant(plaats)
int plaats[8];
{
 screen=pr_open("/dev/fb");
 pr_vector(screen,plaats[0],plaats[1],plaats[2],plaats[3],PIX_NOT(PIX_SRC)&PIX_DST,1);
 pr_vector(screen,plaats[4],plaats[5],plaats[6],plaats[7],PIX_NOT(PIX_SRC)&PIX_DST,1);
 pr_vector(screen,plaats[0],plaats[1],plaats[4],plaats[5],PIX_NOT(PIX_SRC)&PIX_DST,1);
 pr_vector(screen,plaats[2],plaats[3],plaats[6],plaats[7],PIX_NOT(PIX_SRC)&PIX_DST,1);
 pr_close(screen);
}

static void
plaatsvierkant(zijde)
int zijde;
{
   if (zijde==0)
       {
        placel[0]=(int)panel_get(L[afdrukken[te]],PANEL_ITEM_X)-4;
        placel[2]=placel[0]+12+8+9*lengtes[afdrukken[te]][0]-3;
        placel[1]=(int)panel_get(L[afdrukken[te]],PANEL_ITEM_Y)-4+16;
        placel[3]=placel[1];
        placel[4]=placel[0];
        placel[5]=placel[1]+28-2;
        placel[6]=placel[2];
        placel[7]=placel[5];
       }
 else
       {
        placer[0]=(int)panel_get(R[selectie],PANEL_ITEM_X)-4+550;
        placer[2]=placer[0]+12+8+9*lengtes[selectie][1]-3;
        placer[1]=(int)panel_get(R[selectie],PANEL_ITEM_Y)-4+16;
        placer[3]=placer[1];
        placer[4]=placer[0];
        placer[5]=placer[1]+28-2;
        placer[6]=placer[2];
        placer[7]=placer[5];
       }
}

static void
plaatsruit()
{
 int h;
 if (dispansw=='y')
     {
      placew[0]=(int)panel_get(R[ant[afdrukken[te]][1]],PANEL_ITEM_X)-6+550;
      placew[1]=(int)panel_get(R[ant[afdrukken[te]][1]],PANEL_ITEM_Y)+10+16;
      placew[4]=placew[0]+12+12+9*lengtes[ant[afdrukken[te]][1]][1]-3;
      placew[2]=(placew[0]+placew[4])/2;
      placew[3]=placew[1]-20;
      placew[5]=placew[1];
      placew[6]=placew[2];
      placew[7]=placew[1]+20-2;
      h=placew[4];
      placew[4]=placew[6];
      placew[6]=h;
      h=placew[5];
```

```
      placew[5]=placew[7];
      placew[7]=h;
     }
}

static void
begin(item,event)
Panel_item item;
Event *event;
{
 int i;
 if(stop==1)
    {
     window_set(linker,WIN_CONSUME_PICK_EVENT,WIN_MOUSE_BUTTONS,0);
     window_set(rechter,WIN_CONSUME_PICK_EVENT,WIN_MOUSE_BUTTONS,0);
     tekstscherm();
     for(i=0;i<=aantalwoorden;i++)
         ant[i][2]=0;
    }
 else
     panel_set(warning2,PANEL_SHOW_ITEM,TRUE,0);
 stop=0;
}

static void
einde(item,event)
Panel_item item;
Event *event;
{
 stop=1;
 if (beginmaar==1)
     {
      fprintf(fp,"\n%s","stop ");
      tijd();
      fclose(fp);
      window_destroy(frame);
     }
}

static void
naam_proc(item,event)
Panel_item item;
Event *event;
{
 struct timeval tp;
 struct timezone tzp;
 char naam[16],name[15],getal,maxl,*p;
 int n,x;
 long hulp;
 DIR *dirp;
 struct direct *dp;
 strcpy(naam, (char *)panel_get_value(item));
 strcpy(name,naam);
 dirp=opendir("/usr2/rooy/.");
 n=strlen(naam);
 maxl='1';
 for(dp=readdir(dirp);dp!=NULL;dp=readdir(dirp))
     {
      if (strncmp(naam,dp->d_name,n)==0)
          {
           if((p=(char *)strpbrk(dp->d_name,"123456789"))!=NULL)
             {
              getal=(char)((int)*p + 1);
              if (getal>maxl)
```

```
                    maxl=getal;
                }
            }
        }
    naam[n]=maxl;
    naam[n+1]='\0';
    strcpy(uitv,naam);
    closedir(dirp);
    window_destroy(linksonder);
    messagescherm();
    beginmaar=1;
    fp=fopen(uitv,"w");
    fprintf(fp,"%s\n",name);
    fprintf(fp,"%d ",nrrows);
    fprintf(fp,"%d ",nrcols);
    fprintf(fp,"%c ",dispansw);
    fprintf(fp,"%d ",whichalg);
    fprintf(fp,"%c ",contbut);
    if (contbut=='n')
        fprintf(fp,"%d ",secdel);
    fprintf(fp,"%s ",maxtrial);
    fprintf(fp,"%c ",trialbut);
    fprintf(fp,"%c ",stopbut);
    fprintf(fp,"%s\n",inpfilename);
    gettimeofday(&tp, &tzp);
    x=(double)tp.tv_usec/1000000;
    hulp=tp.tv_sec;
    start=tp.tv_sec+x;
    fprintf(fp,"%s %s\n","start",ctime(&hulp));
    Initialisatie();
}

static void
messagescherm()
{
    linksonder=window_create(frame,PANEL,
                             WIN_X,0,
                             WIN_Y,800,
                             WIN_HEIGHT,70,
                             WIN_WIDTH,545,
                             0);
    warning2=panel_create_item(linksonder,PANEL_MESSAGE,
                               PANEL_LABEL_STRING,"eerst stop aanklikken",
                               PANEL_SHOW_ITEM,FALSE,
                               0);
    warning=panel_create_item(linksonder,PANEL_MESSAGE,
                              PANEL_LABEL_STRING,"klik eerst een woord aan,rechts op het be
                              PANEL_SHOW_ITEM,FALSE,
                              0);
    window_fit(linksonder);
}

static void
tekstscherm()
{
    linksonder=window_create(frame,PANEL,
                             WIN_X,0,
                             WIN_Y,800,
                             WIN_HEIGHT,70,
                             WIN_WIDTH,545,
                             0);
    naam_item=panel_create_item(linksonder,PANEL_TEXT,
                                PANEL_LABEL_STRING,"uw achternaam: ",
                                PANEL_VALUE_STORED_LENGTH,15,
```

```
                                PANEL_VALUE_DISPLAY_LENGTH,15,
                                PANEL_NOTIFY_PROC,naam_proc,
                                0);
 window_set(frame,WIN_MOUSE_XY,135,820,0);
 window_fit(linksonder);
}

static void
tijd()
{
 struct timeval tp;
 struct timezone tzp;
 double verbruiktetijd,x;
 gettimeofday(&tp,&tzp);
 x=(double)tp.tv_usec/1000000;
 verbruiktetijd=(tp.tv_sec+x)-start;
 fprintf(fp,"%.2f ",verbruiktetijd);
}

static void
teller_ophoging()
{
 if (trial[2]=='9')
     {
      trial[2]='0';
      if (trial[1]=='9')
          {
           trial[1]='0';
           if (trial[0]=='9')
               einde();
           else
               trial[0]=(char)((int)trial[0]+1);
          }
      else
          trial[1]=(char)((int)trial[1]+1);
     }
 else
     trial[2]=(char)((int)trial[2]+1);
 strcpy(poging,"trial=");
 strcat(poging,trial);
}


Dubbel_rechts(t,woord1)
int t;
char woord1[35];
{
 char woord2[35];
 int i,j,l,teller;
 for(i=1;i<t;i++)
     {
      if (k[i][1]==0)
          {
           Positioneer(k[i][0]);
           for(j=0;getc(fp)!=';';j++);
           for(j=0;(woord2[j]=getc(fp))!='\n';j++);
           woord2[j]='\0';
           if (strcmp(woord1,woord2)==0)
               {
                k[t][1]=1;
                for(j=1;j<=aantalwoorden;j++)
                    {
                     if (ant[j][1]==t)
                         {
```

```
                        teller=0;
                        for(l=1;l<=i;l++)
                            {
                             if (k[l][1]==0)
                                 teller++;
                            }
                        ant[j][1]=teller;
                        }
                    }
                return(-1);
               }
           }
      }
 return(1);
}
```

Bijlage 2.

```
opties.c        Wed Jun  8 15:26:44 1988        1

#include <stdio.h>

explanation()
  { printf("with this program you can give the options for the program\n");
    printf("overhoor.c, which teaches you foreign words.\n");
    printf("the following questions have to be answered:\n");
    printf("    1 number of rows? (0<number of rows<17)\n");
    printf("        choose how many rows of foreign words on your screen.\n");
    printf("    2 number of colums? (0<number of colums<5)\n");
    printf("        choose how many colums of foreign words on your screen.\n");
    printf("    3 give the correct answer? (y(es) or n(o))\n");
    printf("        choose whether you want the program to give the correct\n");
    printf("        answer\n");
    printf("    4 which algorithem? (1,2,3 or 4)\n");
    printf("        1 constantly gives you all the foreign words.\n");
    printf("        2 gives you only the words you had wrong, until\n");
    printf("          the max number of trials is achieved.\n");
    printf("        3 you self can choose the words of which you\n");
    printf("          want to give the translation\n");
    printf("        4 gives you the words with the lowest value\n");
    printf("          each time you give an good answer that word\n");
    printf("          gets an higher value, if you give an wrong\n");
    printf("          answer that word gets value zero\n");
    printf("    5 do you want a continue button? (y or n)\n");
    printf("        if you don't want this button you can choose\n");
    printf("        in the next question how many seconds delay\n");
    printf("        the computer gives you before you can give\n");
    printf("        a translation for the next word, else you can\n");
    printf("        choose your own delay with continue\n");
    printf("    6 how many seconds delay? (0<delay<20)\n");
    printf("    7 max number of trials? (000<maxtrial<=999)\n");
    printf("        choose how many translations can be given\n");
    printf("    8 trialbutton? (y or n)\n");
    printf("        do you want a button on your screen which says\n");
    printf("        how many trials you have had?\n");
    printf("    9 stopbutton? (y or n)\n");
    printf("        if you haven't got a stopbutton the only way\n");
    printf("        to exit overhoor.c is to reach maxtrial\n");
    printf("   10 name of inputfile? (max 20 characters)\n");
    printf("        say which inputfile of foreign words you want to use.\n");
    printf("        this file should be filled in the following structure:\n");
    printf("               foreign word;translation\n");
    printf("               foreign word;translation\n");
    printf("                 ...       ;  ...\n");
    printf("\n");
    printf("\n");
  }

main()
{
    int  nrrows,nrcols,done,whichalg,secdel;
    char dispansw,expl,contbut,trialbut,stopbut;
    char inpfilename[20],maxtrial[3];

    FILE  *fopen(), *fp;
    fp=fopen("options","w");

    system("clear");
    done=0;
    do
     { printf("do you want an explanation about this program?  ");
       scanf("%s",&expl);
       printf("\n");
       if ((expl!='n') && (expl!='y'))
```

```
          { printf("type y or n");
            printf("\n");
          }
        else done=1;
      } while(!done);
    if (expl=='y')
        explanation();

    done=0;
    do
      { printf("number of rows?  ");
        scanf("%d",&nrrows);
        printf("\n");
        if ((nrrows>0) && (nrrows<17))
                  done=1;
        else
           { printf("0<number of rows<17");
             printf("\n");
           }
      } while(!done);
    fprintf(fp,"%d\n",nrrows);

    done=0;
    do
     { printf("number of colums?  ");
       scanf("%d",&nrcols);
       printf("\n");
       if ((nrcols>0) && (nrcols<5))
                 done=1;
       else
          { printf("0<number of colums<5");
            printf("\n");
          }
     } while(!done);
   fprintf(fp,"%d\n",nrcols);

    done=0;
    do
     { printf("give the correct answer?  ");
       scanf("\n");
       scanf("%c",&dispansw);
       printf("\n");
       if ((dispansw!='n') && (dispansw!='y'))
          { printf("type y or n");
            printf("\n");
          }
       else done=1;
     } while(!done);
    fprintf(fp,"%c\n",dispansw);

    done=0;
    do
     { printf("which algorithem?  ");
       scanf("%d",&whichalg);
       printf("\n");
       if ((whichalg>0) && (whichalg<5))
                 done=1;
       else
          { printf(" we have four algorithems");
            printf("\n");
          }
     } while(!done);
   fprintf(fp,"%d\n",whichalg);
```

```
   done=0;
    do
     { printf("do you want a continue button?  ");
       scanf("\n");
       scanf("%s",&contbut);
       printf("\n");
       if ((contbut!='n') && (contbut!='y'))
          { printf("type y or n");
            printf("\n");
          }
       else done=1;
     } while(!done);
   fprintf(fp,"%c\n",contbut);

   if (contbut=='n')
       {
        done=0;
        do
          { printf("how many seconds delay?  ");
            scanf("%d",&secdel);
            printf("\n");
            if ((secdel>0) && (secdel<20))
                  done=1;
            else
                { printf("0<delay<20");
                  printf("\n");
                }
           } while(!done);
        fprintf(fp,"%d\n",secdel);
       }

   printf("max number of trials?  ");
   scanf("%s",maxtrial);
   printf("\n");
   fprintf(fp,"%s\n",maxtrial);

   done=0;
    do
     { printf("trialbutton?  ");
       scanf("\n");
       scanf("%c",&trialbut);
       printf("\n");
       if ((trialbut!='n') && (trialbut!='y'))
          { printf("type y or n");
            printf("\n");
          }
       else done=1;
     } while(!done);
    fprintf(fp,"%c\n",trialbut);

   done=0;
    do
     { printf("stopbutton?  ");
       scanf("\n");
       scanf("%c",&stopbut);
       printf("\n");
       if ((stopbut!='n') && (stopbut!='y'))
          { printf("type y or n");
            printf("\n");
          }
       else done=1;
     } while(!done);
    fprintf(fp,"%c\n",stopbut);
```

```
   printf("name of inputfile?  ");
   scanf("%20s",inpfilename);
   printf("\n");
   fprintf(fp,"%s\n",inpfilename);

   fclose(fp);
   printf("this program has ended now\n");
   printf("\n");
}
```