Simulatie van een verkeersstroom op de autosnelweg

D-95-6A
R.S. Sukdeo & dr. P.H. Polak
Leidschendam, 1995
Stichting Wetenschappelijk Onderzoek Verkeersveiligheid SWOV

Stichting Wetenschappelijk Onderzoek Verkeersveiligheid SWOV

Stichting
Wetenschappelijk
Onderzoek
Verkeersveiligheid
SWOV

Postbus 1090
2260 BB Leidschendam
Duindoorn 32
telefoon 070 3209323
telefax 070-3201261

Inhoud

1.	*Inleiding*	4
2.	*Het onderzoek*	5
2.1.	Doelstelling	5
2.2.	Het model	5
2.2.1.	Input	6
2.2.2.	Het genereren van de gaps	6
2.2.3.	Het bepalen van de rijstrook	6
2.2.4.	Gegevensberekeningen	7
2.2.5.	Output	7
3.	*De resultaten*	9
4.	*Conclusie*	11
Literatuur		12
Bijlagen		13
1.	Programmeren in C	
2.	De listing van het programma	

1. Inleiding

De SWOV doet onder meer onderzoek naar de relatie tussen verkeersveiligheid en verkeersstroomvariabelen op autosnelwegen.
Een verkeersstroom is in het algemeen een ingewikkeld proces. De meest eenvoudige denkbare realistische verkeersstroom is er een die homogeen en stationair is op een eindeloze éénrichtingsweg. Homogeen betekent in dit geval dat de weg geen kruisingen mag hebben. In het algemeen is een verkeersstroom verre van homogeen en stationair.

In zo'n verkeersstroommodel kunnen de voertuigen van hetzelfde type zijn, maar de bestuurders moeten niet allemaal even snel rijden.
Het eenvoudigste model van mogelijke realistische verkeersstromen zal grondig moeten worden onderzocht en op grond hiervan kunnen meer gecompliceerde situaties onderzocht worden.
Eerst worden de variabelen die de stroom adequaat omschrijven, vastgesteld. Deze variabelen moeten een theoretische basis hebben, maar het moet ook mogelijk zijn om ze te controleren met metingen. Er worden gemiddelde waarden gebruikt, omdat metingen die betrekking hebben op een enkele auto op een bepaald moment, ons heel weinig vertellen over de verkeersstroom waartoe de auto behoort. Er kan een gemiddelde worden genomen van een aantal voertuigen, of over een bepaalde tijd of afstand.

Om een theorie, model of simulatierun van verkeersstromen te vergelijken met de werkelijkheid moet worden bepaald welke variabelen relevant zijn en hoe de waarde van deze variabelen door metingen moet worden verkregen uit de werkelijkheid.

Er zijn enige resultaten analytisch afgeleid voor een simpel model van een homogene, stationaire verkeersstroom op een éénrichtingsrijbaan met twee rijstroken. Deze resultaten zijn overeenkomstig de gegevens uit het werkelijke verkeer, in omstandigheden waar interactie tussen auto's niet leidt tot veranderingen in de snelheid (isoveloxisch regime). De analyse moet worden voortgezet voor meer realistische verkeersstromen, zoals verkeersstromen met meer dan twee verschillende snelheden, voertuigen met lengte, auto's die gas afnemen bij een drie-voertuigeninteractie en gapverdelingen met een minimale afstand. Het is onwaarschijnlijk dat voor alle gevallen analytische resultaten kunnen worden verkregen. Daar zal computersimulatie als enige voor noodzakelijk zijn.

2. Het onderzoek

2.1. Doelstelling

Het doel is het ontwikkelen van een simulatiemodel van een homogene, stationaire verkeersstroom op een éénrichtingsrijbaan met twee rijstroken van een autosnelweg.

Een simulatiemodel is een beschrijving van de elementen van een systeem en hun onderlinge relaties, dat op grond hiervan een werkelijk systeem kan nabootsen.
Eerst zal met dit model de analytisch verkregen resultaten vergeleken worden (volgafstandsverdeling en snelheden als functie van de dichtheid, alsmede aantallen inhaalbewegingen en kritieke situaties, alles voor een verkeersstroom op twee rijstroken). Dit dient als controle van de theorie en als controle van het simulatiemodel.
Vervolgens zal het simulatiemodel dienen om meer ingewikkelde situaties of meer realistische verkeersstromen van meer dan twee rijstroken door te rekenen.

Vóór het ontwikkelen van het model is het zinvol om eerst een stroomschema te maken. Het stroomschema geeft een globaal beeld van het model. Het is ook praktisch bij het vertalen van het model naar een computerprogramma.

Om het verkeer op autosnelwegen te kunnen regelen is het noodzakelijk de toestandsvergelijkingen van de verkeersstroom te kennen. Daarvoor moet je eerst bepalen wat de relevante variabelen zijn die de stroom beschrijven. Ook moet je die kunnen meten.
Om een succesvolle afloop te bewerkstelligen bij het modelleren van een systeem is het van belang dat het aantal variabelen beperkt is. Daarom is het aanbrengen van vereenvoudigingen in het algemeen zinvol.

2.2. Het model

We beginnen met de simpelste verkeersstroom die realistisch is: een homogene, stationaire stroom op een éénrichtingsrijbaan met twee rijstroken. Op die rijbaan, met twee rijstroken voor verkeer in dezelfde richting, bevinden zich twee verkeersstromen. Er is een verkeersstroom langzame voertuigen ('vrachtauto's'), die allemaal even hard rijden (altijd op de rechterrijstrook). En er is een verkeersstroom voertuigen ('personenauto's') die sneller is, maar ook allemaal even snel (ze rijden rechts behalve als ze een vrachtauto genaderd zijn tot de afstand L_a).
We beschouwen de voertuigen als punten, de lengte van de voertuigen wordt verwaarloosd.

De stroom personenauto's is een Poissonproces. De stroom vrachtauto's kan een Poissonproces zijn, maar kan ook regelmatig zijn, om resultaten mogelijk te maken, die met analytisch verkregen gegevens vergeleken kunnen worden. Als beide stromen onafhankelijke Poissonprocessen zijn dan vormen de ruimten tussen de voertuigen in elke stroom een

exponentiële verdeling, met de dichtheden (dichtheid = het aantal voertuigen per meter) als parameters. L is de totale invloedslengte, L_a is de invloedslengte achter een vrachtauto en L_v is de invloedslengte vóór een vrachtauto. De vrachtauto's rijden altijd op de rechterrijstrook, terwijl de personenauto's rechts rijden tenzij ze te dicht (minder dan L_a meter) achter een vrachtauto komen. Ze wisselen van rijstrook en rijden links totdat ze een opening op de rechterrijstrook zien die langer is dan L meter, dan wisselen ze weer naar rechts, op een plaats van L_v meter voor de vrachtauto.

2.2.1. *Input*

De gegevens die aan het begin moeten worden ingevoerd zijn:
- De verdeling van de gaps tussen de vrachtauto's (exponentieel of regelmatig)
- De dichtheid van de personenauto's (afgekort: *dp*);
- De dichtheid van de vrachtauto's (afgekort: *dv*);
- De invloedslengte vóór een vrachtauto;
- De invloedslengte achter een vrachtauto;
- De snelheid van de personenauto's;
- De snelheid van de vrachtauto's;
- De lengte van de personenauto's;
- De lengte van de vrachtauto's;
- Het totaal aantal voertuigen (als er is aangegeven dat het programma bij dit aantal moet stoppen);
- De naam van een bestand (als de uitvoer naar een bestand moet worden geschreven);

2.2 2. *Het genereren van de gaps*

Bij het Poissonproces doen we trekkingen uit de exponentiële verdeling. Deze heeft als cumulatieve verdelingsfunctie $F(x)=1-e^{\lambda x}$, waarbij λ de dichtheid is en x de gap. Om de gaps te genereren maken we gebruik van toevalsgetallen. Een toevalsgetal is een willekeurig getal tussen de nul en de één. Om x uit te drukken als een functie van een toevalsgetal m stellen we m gelijk aan de verdelingsfunctie.

Dus $m = 1-e^{-\lambda x} \Rightarrow e^{-\lambda x}+m = 1 \Rightarrow e^{-\lambda x} = 1-m \Rightarrow -\lambda x = ln(1-m) \Rightarrow x = -ln(1-m)/\lambda$.

Omdat $1-m$ ook een getal is tussen de nul en één, kunnen we ook schrijven $x = -ln(m)/\lambda$. Hierbij mag m geen nul zijn, omdat $ln(0)$ niet bestaat.
Bij de regelmatige verdeling is $x = 1/\lambda$.

2.2.3. *Het bepalen van de rijstrook*

We beginnen met twee vrachtauto's en één personenauto. De vrachtauto met de lage passeertijd, heeft bij de start van het programma de tijd 0. De vrachtauto met de hoge passeertijd passeert op de tijd van de eerste gap tussen de vrachtauto's. De personenauto passeert op de tijd van de eerste gap tussen de personenauto's.

Als de passeertijd van de personenauto niet tussen de passeertijden van de beide vrachtauto's ligt, dan worden er net zolang gaps (en vrachtauto's) gegenereerd (de vrachtauto met de lage passeertijd krijgt steeds de passeertijd van de andere vrachtauto), totdat de passeertijd van de personenauto wel tussen de passeertijden van de twee vrachtauto's ligt. Zodra dit het geval is, wordt er gekeken of één van de afstanden tussen de personenauto en de vrachtauto's binnen het invloedsgebied ligt.
Als de afstand tussen de personenauto en een van de twee vrachtauto's kleiner is dan een van de invloedslengtes, dan weet je dat de personenauto op de linkerrijstrook rijdt. Als de afstand groter is, dan rijdt de personenauto rechts. Hierna wordt er een nieuwe gap en tijd gegenereerd voor de personenauto. Er wordt weer gekeken of de tijd van de personenauto wel of niet tussen de tijden van de vrachtauto's ligt.

Het is ook mogelijk dat de passeertijd van de personenauto gelijk is aan de passeertijd van één van de vrachtauto's. Bij die situatie moet de personenauto natuurlijk links rijden.

Als je wilt dat het programma bij een bepaald aantal voertuigen stopt, kan je dit aan het begin van het programma aangeven. Anders wordt het proces net zolang herhaald tot er een toets wordt ingedrukt. Het programma wordt dan onderbroken en je hebt de mogelijkheid om te stoppen door op Enter te drukken of verder te gaan door een ander willekeurige toets in te drukken.

2.2.4. *Gegevensberekeningen*

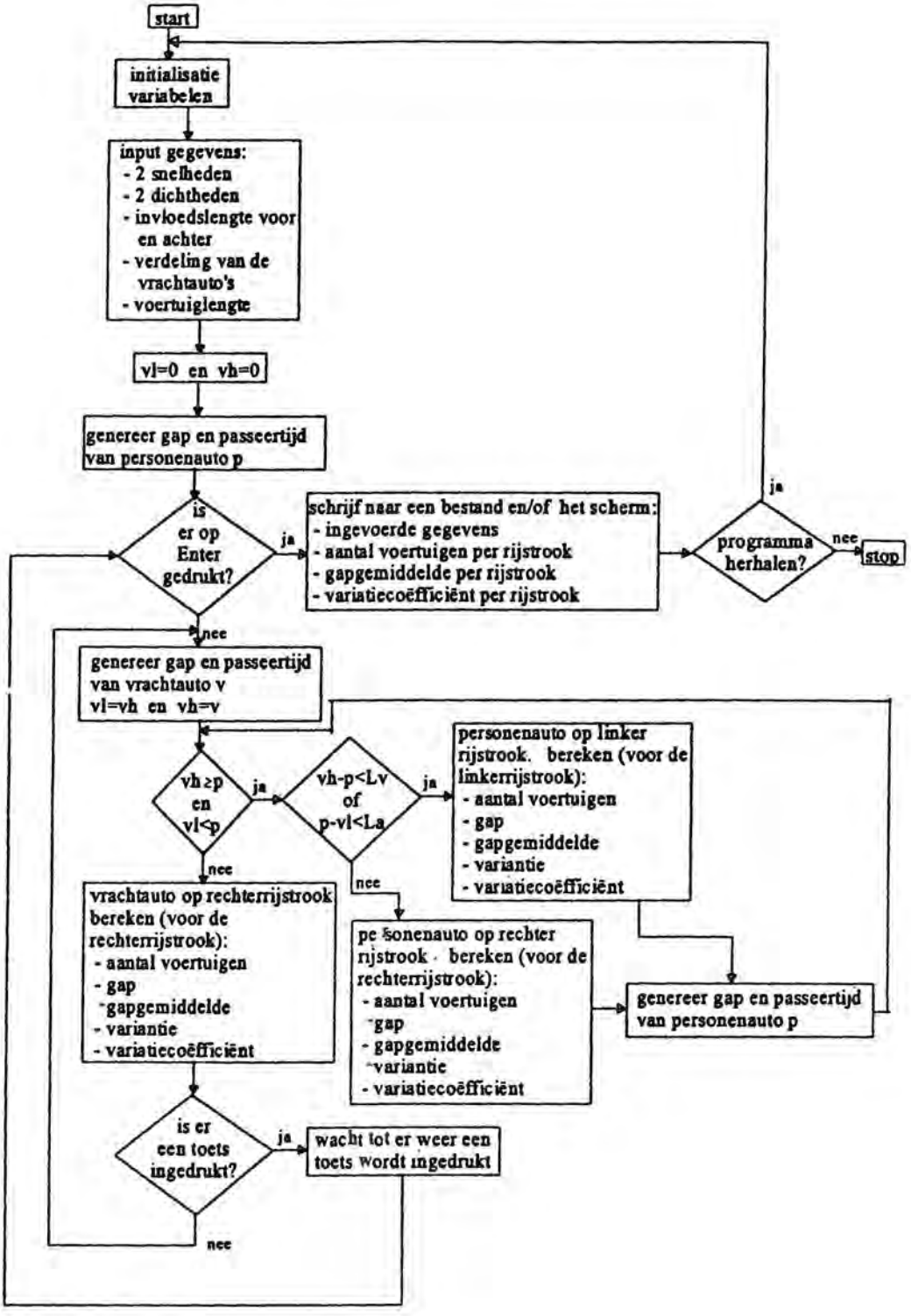
Vóór het genereren van een nieuwe gap en tijd wordt er *per rijstrook* steeds een aantal berekeningen gemaakt, namelijk:
- Het totaal aantal voertuigen;
- De gap (de tijd minus de tijd van het voorgaand voertuig op dezelfde strook);
- Het gapgemiddelde (de tijd gedeeld door het aantal voertuigen);
- De som van de gekwadrateerde gaps;
- Het kwadraat van het gemiddelde van de gaps (het kwadraat van het gapgemiddelde);
- De variantie (de som van de gekwadrateerde gaps gedeeld door het aantal voertuigen op de desbetreffende strook minus het kwadraat van het gemiddelde van de gaps);
- De variatiecoëfficiënt (de wortel van de variantie gedeeld door het gapgemiddelde);

2.2.5. *Output*

De volgende gegevens worden naar het scherm geschreven en ook naar een bestand als daar voor wordt gekozen:
- Aantal voertuigen;
- De passeertijd;
- De gap op de rijstrook;
- De gemiddelde gap per rijstrook;
- De variatiecoëfficiënt per rijstrook.

Aan het einde van het programma worden de gegevens die werden ingevoerd nog een keer gegeven evenals het aantal voertuigen, het gap-gemiddelde en de variatiecoëfficiënt per rijstrook.
Hierna wordt gevraagd om herhaling van het programma.

Hieronder is het stroomschema afgebeeld.
Zie *Bijlage 1* voor een introductie van programmeren in C. Zie *Bijlage 2* voor de listing van het computerprogramma.



3. De resultaten

Voor een vergelijking met de analytisch berekende resultaten hebben we het programma laten lopen voor een aantal situaties:

1. De verdeling van de gaps tussen de vrachtauto's is exponentieel en $L_v = L_a = 0,5$.
   $dp = dv = 0,1 \ldots 2,0$ (stapgrootte 0,1)
2. De verdeling van de gaps tussen de vrachtauto's is exponentieel en $L_v = 1$, $L_a = 0$.
   $dp = dv = 0,1 \ldots 2,0$ (stapgrootte 0,1)
3. De verdeling van de gaps tussen de vrachtauto's is regelmatig en $L_v = L_a = 0,5$.
   $dp = dv = 0,1 \ldots 1,0$ (stapgrootte 0,1)

Het totaal aantal voertuigen was bij iedere run 10.000.

Situatie 1

		variatiecoëfficiënt	
dp	dv	links	rechts
0,1	0,1	1,114	0,963
0,2	0,2	1,090	0,901
0,3	0,3	1,105	0,870
0,4	0,4	1,115	0,854
0,5	0,5	1,124	0,828
0,6	0,6	1,162	0,823
0,7	0,7	1,155	0,804
0,8	0,8	1,097	0,785
0,9	0,9	1,175	0,787
1,0	1,0	1,152	0,775
1,1	1,1	1,120	0,768
1,2	1,2	1,138	0,785
1,3	1,3	1,138	0,769
1,4	1,4	1,112	0,769
1,5	1,5	1,114	0,766
1,6	1,6	1,131	0,787
1,7	1,7	1,133	0,784
1,8	1,8	1,106	0,797
1,9	1,9	1,118	0,789
2,0	2,0	1,150	0,785

Situatie 2

		variatiecoëfficiënt	
dp	dv	links	rechts
0,1	0,1	1,024	0,953
0,2	0,2	1,032	0,926
0,3	0,3	1,127	0,888
0,4	0,4	1,186	0,871
0,5	0,5	1,162	0,852
0,6	0,6	1,151	0,879
0,7	0,7	1,135	0,868
0,8	0,8	1,169	0,842
0,9	0,9	1,164	0,855
1,0	1,0	1,146	0,851
1,1	1,1	1,159	0,857
1,2	1,2	1,172	0,866
1,3	1,3	1,140	0,859
1,4	1,4	1,216	0,876
1,5	1,5	1,142	0,883
1,6	1,6	1,127	0,874
1,7	1,7	1,112	0,886
1,8	1,8	1,099	0,874
1,9	1,9	1,112	0,885
2,0	2,0	1,108	0,895

Situatie 3

		variatiecoëfficiënt	
dp	dv	links	rechts
0,1	0,1	1,025	0,627
0,2	0,2	0,972	0,588
0,3	0,3	1,002	0,543
0,4	0,4	0,996	0,507
0,5	0,5	0,985	0,466
0,6	0,6	1,016	0,411
0,7	0,7	0,975	0,361
0,8	0,8	1,008	0,302
0,9	0,9	1,009	0,214
1,0	1,0	1,020	0,000

Voor situatie 1 en 2 zijn de resultaten analytisch berekend:

Situatie 1

		variatiecoëfficiënt
dp	dv	rechts
0,1	0,1	0,95365
0,2	0,2	0,91417
0,3	0,3	0,88097
0,4	0,4	0,85343
0,5	0,5	0,83099
0,6	0,6	0,81309
0,7	0,7	0,79920
0,8	0,8	0,78883
0,9	0,9	0,78153
1,0	1,0	0,77687
1,1	1,1	0,77448
1,2	1,2	0,77401
1,3	1,3	0,77517
1,4	1,4	0,77768
1,5	1,5	0,78131
1,6	1,6	0,78586
1,7	1,7	0,79115
1,8	1,8	0,79701
1,9	1,9	0,80333
2,0	2,0	0,80998

Situatie 2

		variatiecoëfficiënt
dp	dv	rechts
0,1	0,1	0,95590
0,2	0,2	0,92228
0,3	0,3	0,89728
0,4	0,4	0,87928
0,5	0,5	0,86686
0,6	0,6	0,85884
0,7	0,7	0,85426
0,8	0,8	0,85232
0,9	0,9	0,85240
1,0	1,0	0,85401
1,1	1,1	0,85675
1,2	1,2	0,86034
1,3	1,3	0,86453
1,4	1,4	0,86915
1,5	1,5	0,87405
1,6	1,6	0,87914
1,7	1,7	0,88432
1,8	1,8	0,88952
1,9	1,9	0,89471
2,0	2,0	0,89984

Na vergelijking van de resultaten van het simulatiemodel met de analytische berekende resultaten, is gebleken dat de verschillen tussen beide resultaten niet meer dan 2% zijn.
Dit percentage valt binnen de grenzen die te verwachten zijn bij resultaten van simulaties op basis van 10.000 gevallen.

4. Conclusie

Een eerste versie van een model van een verkeersstroom is ontwikkeld. Deze versie moest aan een aantal eisen voldoen, namelijk:
- De weg is een éénrichtingsrijbaan;
- De weg is oneindig;
- De voertuigen hebben nog geen lengte, ze worden als punten beschouwd;
- Er zijn twee typen voertuigen, snelle en langzame;
- De snelle auto's rijden allemaal met dezelfde snelheid en de langzame auto's rijden ook allemaal met dezelfde snelheid;
- De auto's veranderen nooit van snelheid, ze rijden met een constante snelheid;
- Er zijn twee rijstroken;
- De langzame auto's rijden altijd op de rechterrijstrook;
- De snelle auto's rijden ook op de rechterrijstrook, behalve als ze inhalen, dan rijden ze op de linkerrijstrook.

Het model is naar een computerprogramma vertaald. Het computerprogramma werkt en voldoet aan de eisen die er tot nu toe aan gesteld werden.
De resultaten zijn vergeleken met de analytische gegevens. Helaas heeft het computerprogramma bij veel voertuigen ook veel rekentijd nodig. Voor een run van 10.000 voertuigen heeft een 386 machine al gauw tien minuten nodig.

Nu het simpelste model goed werkt, kan het model en het programma worden uitgebreid.
Als de verkeersstroom realistischer moet zijn, dan moet er worden uitgebreid, op onder andere de volgende aspecten:
- De lengte van de voertuigen moet groter dan nul zijn. De gaps zullen dan ook anders moeten worden berekend. Er kan worden begonnen met twee verschillende lengtes.
- Drie-voertuigeninteractie: als de afstand achter een langzamere auto minder is dan L_a meter, dan wordt er alleen van strook gewisseld als dit mogelijk is (er kan onvoldoende ruimte zijn op de linkerrijstrook als twee snelle auto's op de linkerrijstrook dicht op elkaar rijden). Anders moet de snellere auto zijn snelheid minderen tot de snelheid van de langzamere auto en wachten tot het wel mogelijk is om in te halen.
- Meer dan twee verschillende snelheden, omdat ook dit realistischer is.
- Meer dan twee rijstroken, omdat we in Nederland ook autosnelwegen hebben met drie en vier rijstroken.

Een uitgebreide versie van het programma moet al deze aspecten bevatten.

Literatuur

Buijs, A. (1991). *Statistiek om mee te werken.* Stenfert Kroese, Leiden.

Erlenkötter, H. & Reher, V. (1992). *Programmeren in C.* Het Spectrum, Utrecht.

Griep, P.A.M. & Flapper, S.D.P. (1987). *Discrete simulatie.* Academic Service, Schoonhoven.

Kettenis, D.L. (1990). *Simulatie.* Stenfert Kroese, Leiden.

Polak, P.H. (1995). *Traffic Flow Theory derived from First Principles; The development of a causal theory which encompasses safety.* SWOV, Leidschendam. [nog niet gepubliceerd]

Vliegenthart, A.C. (1993). *Leerboek Operations Research.* Academic Service, Schoonhoven.

Winston, W.L. (1987). *Operations Research; Applications and Algorithms.* PWS-Kent, Boston.

Bijlage 1 Programmeren in C

Include
In bijna elk C-programma staan aan het begin één of meer *include*-opdrachten. *Include* is geen instructie, maar een richtlijn voor de pre-processor. Voordat de brontekst wordt gecompileerd, zoekt de pre-processor naar regels met teken # (hekje of matje). Opdrachten achter dit teken zijn bedoeld voor de pre-processor. Het woord *include* betekent dat het bestand dat achter tussen de punthakenstaat, ingevoegd moet worden in de brontekst. Het compiler-pakket omvat een dertigtal *include*-bestanden, te herkennen aan de extensie *.h*. Ze hebben onder meer tot doel een programma te voorzien van de definities die nodig zijn voor de gebruikte functies.

Main
Het woord *main* komt voor in ieder zelfstandig C-programma. Vanaf dat punt wordt het programma uitgevoerd. De ronde haken achter het woord *main* is kenmerkend voor de C-taal, want een C-programma bestaat geheel uit functies. Tussen de haken staan de waarden (argumenten) waarmee een functie werkt. Als er niets tussen de haken staat, dan krijgt de functie geen argumenten mee.

Toelichtende tekst en commentaar staat tussen de codes /* en */. De compiler negeert bij het vertalen alles wat tussen deze codes staat.
De accolades markeren het begin en het eind van een functie of een groep opdrachten die bij elkaar hoort (een blok).
Elke opdracht eindigt met een puntkomma.

Functies
Eén van de meest gebruikte functies is *printf*. De tekst die moet worden weergegeven staat tussen dubbele aanhalingstekens. Tussen deze tekens staan ook codes: de code voor een nieuwe regel is \n. De functie *printf* kan behalve tekst uitvoeren ook expressies evalueren.
Het procentteken met een letter er achter (bijv. *%d*), waarbij de letter betrekking heeft op het soort variabele, is een code die een plaats reserveert in de uitvoer.
De uitvoer wordt naar een stream geschreven als je *fprintf* gebruikt. Een stream is een reeks tekens. Waar de reeks tekens vandaan komt of heen gaat, hangt af van de aard van het geopende bestand. Het besturings-systeem beschouwt de randapparaten ook als een bestand.
De functie *scanf* laadt gegevens. Net als bij *printf* is het eerste argument een string met reserveringscode voor de in te voeren variabele, en het tweede argument de waarde of variabele zelf. De code & levert het geheugenadres van de variabele.
De functie *fopen* opent een bestand. Daarna is het mogelijk in dat bestand te lezen en te schrijven. Na afloop kan je het bestand weer sluiten met de functie *fclose*.
Als een bestand niet geopend kan worden dan is de functiewaarde nul (in C gedefinieerd als NULL).
De functie *getch* (get character) laadt een teken rechtstreeks uit het toetsenbord. De functiewaarde is de code van het teken. Het programma gaat verder zodra er een toets is ingedrukt.

De functie *kbhit* (keyboard hit) controleert of er een toets is ingedrukt.
In C kan je gebruik maken van de functie *rand* om toevalsgetallen te genereren. Er worden dan getallen gegenereerd tussen de 0 en RAND_MAX. RAND_MAX is de maximale waarde die kan worden gegenereerd.

Programmablokken herhalen
In een programma kunt u een standaardbewerking voortdurend laten herhalen door die bewerking op te nemen in een lus. Het programma wordt niet zonder meer uitgevoerd van het begin tot het einde, maar na afloop van een gemarkeerd blok opdrachten, springt het programma terug naar het begin van dit blok en voert het die opdrachten nog een of meer keren uit. Het programma moet weten welk deel moet worden herhaald en hoe vaak dat moet gebeuren. Het blok van een lus schrijven we tussen accolades. Zodra niet meer voldaan wordt aan de voorwaarde gaat het programma verder op de eerste regel na het blok tussen de accolades.
Een *for*-lus is de beste lus voor gevallen waarin vooraf bekend is hoe vaak de lus moet worden uitgevoerd. Met de instructie *for* geeft u op hoe vaak de lus moet worden herhaald. Het programma telt het aantal malen dat de lus is doorlopen met behulp van een variabele die bij elke doorloop wordt opgehoogd of verlaagd.
Bij een *while*-lus is vooraf niet vastgesteld hoe vaak de lus moet worden doorlopen. Er is daarentegen een voorwaarde waaraan in het begin moet zijn voldaan om de lus uit te voeren.
Bij een *do...while*-lus test het programma aan het einde van de lus of de lus nog een keer moet worden uitgevoerd of niet. Het programma doorloopt de lus dus ten minste eenmaal.

Vertakkingen
Bij de instructie *if* staat het opdrachtenblok ook tussen accolades. Dit blok wordt alleen uitgevoerd als de voorwaarde achter *if* waar is.
De instructie *if* kan worden uitgebreid met de optie *else*, als er een keuze is uit twee mogelijkheden. Als de uitdrukking achter *if* waar is, worden de opdrachten na *if* uitgevoerd, anders die achter *else*.

Vergelijkingsoperatoren
Om twee waarden te vergelijken gebruik je vergelijkingsoperatoren. Een vergelijkingsoperator geeft aan hoe de twee waarden moeten worden vergeleken. C kent onder andere de volgende vergelijkingsoperatoren:
< kleiner dan
> groter dan
<= kleiner dan of gelijk aan
>= groter dan of gelijk aan
!= ongelijk aan
== gelijk aan

Logische operatoren
Als twee uitspraken tegelijk moeten gelden, dan gebruik je de logische operator AND. In C wordt de logische operator AND geschreven met de code *&&*.
Als het voldoende is als aan één van de twee uitspraken wordt voldaan, gebruik je de logische operator OR. In C wordt de logische operator OR geschreven met code ||.

De logische operator NOT, die de waarde van een logische uitdrukking verandert in het tegendeel, wordt geschreven als de code !.

Om naar een andere plaats in het programma te springen wordt de instructie *goto* gebruikt. Het doel van een sprong wordt gemarkeerd door een label.

De variabelen
De voornaamste typen van variabelen en de grootte daarvan zijn:

char	: character	1 byte
int	: integer (gehele getallen)	2 bytes
long	: long integer	4 bytes
float	: breuken en decimale getallen	4 bytes
double	: long float	8 bytes

In het programma worden de volgende variabelen gebruikt:

Integer variabelen:
sp = snelheid van de personenauto's
sv = snelheid van vrachtauto's

lengte1 = lengte van de personenauto's
lengte2 = lengte van de vrachtauto's

verdlvr = soort verdeling van de afstanden tussen de vrachtauto's

f = hulpvariabele bij een keuzevraag
u = hulpvariabele bij een keuzevraag
g = hulpvariabele bij een keuzevraag

Long integer variabelen:
tr = aantal voertuigen op de rechterrijstrook
tl = aantal voertuigen op de linkerrijstrook

aantal = het totaal aantal voertuigen

Character variabelen:
strnr_p = strooknummer van de personenauto
strnr_v = strooknummer van de vrachtauto

ch = hulpvariabele bij het onderbreken van het programma

bestand = naam van het bestand waar de uitvoer naar moet worden geschreven

Pointer-variabelen:
r = schrijft output naar een bestand
s = schrijft output naar het bestand data
t = schrijft output naar het scherm
a = schrijft output naar het bestand links
b = schrijft output naar het bestand rechts

Double variabelen:
gap_p = gap tussen twee personenauto's
gap_v = gap tussen twee vrachtauto's

p = passeertijd personenauto
v = passeertijd vrachtauto

vh = vrachtauto hoog
vl = vrachtauto laag

L_v = invloedslengte voor een personenauto
L_a = invloedslengte achter een vrachtauto

dp = dichtheid van de personenauto's
dv = dichtheid van de vrachtauto's

voorgpass1 = passeertijd voorgaand voertuig op de linkerrijstrook
voorgpass2 = passeertijd voorgaand voertuig op de rechterrijstrook

gapstr = gap op de betreffende strook

variantie = variantie van de gaps

varco1 = variatiecoëfficiënt op de linkerrijstrook
varco2 = variatiecoëfficiënt op de rechterrijstrook

gemgap1 = gemiddelde gap op de linkerrijstrook
gemgap2 = gemiddelde gap op de rechterrijstrook

somgapkwad1 = de som van de gekwadrateerde gaps op de linkerrijstrook
somgapkwad2 = de som van de gekwadrateerde gaps op de rechterrijstrook

kwadgemgap = het kwadraat van het gemiddelde van de gaps op de betreffende strook

x = hulpvariabele bij het berekenen van de variantie

m = randomgetal bij exponentiële verdeling

Bijlage 2 De listing van het programma

```
/*verkeersstroomtheorie*/

/*opdrachten voor de preprocessor*/
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>

/*declaratie pointers*/
FILE *r;
FILE *s;
FILE *t = stdout;
FILE *a;
FILE *b;

/*programma uitvoeren*/
main()
{
/*declaratie variabelen*/
  int sp,sv,lengte1,lengte2,verdlvr,f,u,g;
  long tr,tl,aantal;
  char strnr_p,strnr_v,ch,bestand[14];
  double
gap_p,gap_v,p,v,vh,vl,Lv,La,dp,dv,voorgpass1,voorgpass2,gapstr
,

variantie,varco1,varco2,gemgap1,gemgap2,somgapkwad1,somgapkwad
2,
            kwadgemgap,x,m;


/*de bestanden data, links en rechts openen*/
  s=fopen("data","w");
  a=fopen("links","w");
  b=fopen("rechts","w");

  do
  {
/*maak het scherm leeg*/
  clrscr();
/*uitvoer inleidende tekst naar het scherm*/
  fprintf(t,"Dit programma simuleert een verkeerstroom op een
eenrichtingsrijbaan\n");
  fprintf(t,"met twee rijstroken. Op de rijbaan bevinden zich
twee verkeersstromen:\n");
  fprintf(t," - de vrachtauto's die allemaal even hard rijden
en altijd op de\n");
  fprintf(t,"   rechterrijstrook\n");
  fprintf(t," - de personenauto's die sneller zijn, maar ook
allemaal even snel, ze\n");
  fprintf(t,"   rijden rechts, behalve als ze een vrachtauto
genaderd zijn tot de\n");
  fprintf(t,"   afstand La. Dan gaan ze links en blijven links
tot ze er Lv meter\n");
  fprintf(t,"   voorbij zijn, waarna ze weer naar rechts
gaan.\n");
  fprintf(t,"De stromen zijn onafhankelijke processen met elk
```

```
/*verkeersstroomtheorie*/

/*opdrachten voor de preprocessor*/
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>

/*declaratie pointers*/
FILE *r;
FILE *s;
FILE *t = stdout;
FILE *a;
FILE *b;

/*programma uitvoeren*/
main()
{
/*declaratie variabelen*/
  int sp,sv,lengte1,lengte2,verdlvr,f,u,g;
  long tr,tl,aantal;
  char strnr_p,strnr_v,ch,bestand[14];
  double
gap_p,gap_v,p,v,vh,vl,Lv,La,dp,dv,voorgpass1,voorgpass2,gapstr
,

variantie,varco1,varco2,gemgap1,gemgap2,somgapkwad1,somgapkwad
2,
           kwadgemgap,x,m;


/*de bestanden data, links en rechts openen*/
  s=fopen("data","w");
  a=fopen("links","w");
  b=fopen("rechts","w");

  do
  {
/*maak het scherm leeg*/
  clrscr();
/*uitvoer inleidende tekst naar het scherm*/
  fprintf(t,"Dit programma simuleert een verkeerstroom op een
ΘΘnrichtingsrijbaan\n");
  fprintf(t,"met twee rijstroken. Op de rijbaan bevinden zich
twee verkeersstromen:\n");
  fprintf(t," - de vrachtauto's die allemaal even hard rijden
en altijd op de\n");
  fprintf(t,"   rechterrijstrook\n");
  fprintf(t," - de personenauto's die sneller zijn, maar ook
allemaal even snel, ze\n");
  fprintf(t,"   rijden rechts, behalve als ze een vrachtauto
genaderd zijn tot de\n");
  fprintf(t,"   afstand La. Dan gaan ze links en blijven links
tot ze er Lv meter\n");
  fprintf(t,"   voorbij zijn, waarna ze weer naar rechts
gaan.\n");
  fprintf(t,"De stromen zijn onafhankelijke processen met elk
```

```
als parameter dichtheid\n");
  fprintf(t,"(het aantal voertuigen per meter) - De afstanden
tussen de personenauto's\n");
  fprintf(t,"zijn exponentieel verdeeld. Voor de afstanden
tussen de vrachtauto's kan\n");
  fprintf(t,"je kiezen tussen de regelmatige(1) en de
exponentiële(2) verdeling.\n");

/*invoer gegevens*/
  fprintf(t,"Toets 1 of 2 in voor de bijbehorende keuze van de
verdeling: ");
  scanf("%d",&verdlvr);
  clrscr();
  fprintf(t,"Geef de dichtheid van de personenauto's        :
");
  scanf("%lf",&dp);
  fprintf(t,"Geef de dichtheid van de vrachtauto's          :
");
  scanf("%lf",&dv);
  fprintf(t,"\nGeef de invloedslengte voor een vrachtauto   :
");
  scanf("%lf",&Lv);
  fprintf(t,"Geef de invloedslengte achter een vrachtauto :
");
  scanf("%lf",&La);
  fprintf(t,"\nGeef de snelheid van de personenauto's
: ");
  scanf("%d",&sp);
  fprintf(t,"Geef de snelheid van de vrachtauto's           :
");
  scanf("%d",&sv);
  fprintf(t,"\nGeef de lengte van de personenauto's : ");
  scanf("%d",&lengte1);
  fprintf(t,"Geef de lengte van de vrachtauto's    : ");
  scanf("%d",&lengte2);
  fprintf(t,"\nWilt u dat het programma loopt tot een bepaald
aantal voertuigen?\n");
  fprintf(t,"(1=Ja  2=Nee) : ");
  scanf("%d",&f);
  if (f==1)
      {
      fprintf(t,"Geef het totaal aantal voertuigen : ");
      scanf("%ld",&aantal);
      }
  fprintf(t,"\nWilt u dat de uitvoer naar een bestand worden
geschreven? (1=Ja 2=Nee) : ");
  scanf("%d",&u);
  if (u==1)
      {
      fprintf(t,"Geef de naam van het bestand : ");
      scanf("%s",bestand);
/*het bestand openen*/
      if((r=fopen(bestand,"w"))==NULL)
          fprintf(t,"\nHet bestand %s kon niet worden
geopend!",bestand);
      }
  fprintf(t,"\nDruk een willekeurige toets in om het programma
te laten lopen.\n");
```

```
  fprintf(t,"Druk weer een willekeurige toets in om het
programma te onderbreken.\n");
  fprintf(t,"Druk op Enter om te stoppen.");
/*druk een toets in om met het programma door te gaan*/
  getch();
  tr=0; tl=0; v=0; p=0; vh=0; vl=0; voorgpass1=0;
voorgpass2=0;
  somgapkwad1=0; somgapkwad2=0;

/*initialiseer de toevalsgetallen met een toevalswaarde*/
  randomize();
/*genereer een toevalsgetal tussen 0 en 1*/
  do
      m=(double)rand()/RAND_MAX;
  while (m==0);   /*het toevalsgetal mag geen 0 zijn*/
/*genereer gap en passagetijd van de eerste personenauto*/
  gap_p=-log(m)/dp;
  p=p+gap_p;

  clrscr();
/*uitvoer naar een bestand*/
  if (u==1)
      {
      fprintf(r,"                          Gap
  Gapgemiddelde    Variatieco. \n");
      fprintf(r,"  Totaal  Tijd   Links Rechts  Voertuig
Strook  Links  Rechts   Links  Rechts\n");

fprintf(r,"------------------------------------------------
----------------------------\n");
      }
/*uitvoer naar het scherm*/
  fprintf(t,"                          Gap
Gapgemiddelde    Variatieco. \n");
  fprintf(t,"  Totaal  Tijd   Links Rechts  Voertuig Strook
Links  Rechts   Links  Rechts\n");

fprintf(t,"------------------------------------------------
----------------------------\n");
/*zolang er niet op Enter wordt gedrukt*/
  do
      {
/*als de verdeling regelmatig is*/
      if(verdlvr==1)
          gap_v=(1/dv);
/*als de verdeling exponentieel is*/
      else
          {
/*genereer een toevalsgetal tussen 0 en 1*/
          do
            m=(double)rand()/RAND_MAX;
          while (m==0);   /*het toevalsgetal mag geen 0 zijn*/
/*genereer gap en passagetijd van een vrachtauto*/
          gap_v=-log(m)/dv;
          }
      v=v+gap_v;
      vl=vh;
      vh=v;
```

```
/*zolang p tussen vh en vl ligt of gelijk is aan vh*/
      while (vl<p && p<=vh)
           {
/*als p in het invloedsgebied ligt*/
          if ((vh-p)<La || (p-vl)<Lv)
            {
/*p op de linkerrijstrook*/
            strnr_p='1';
/*het aantal voertuigen op de linkerrijstrook wordt met ΘΘn
verhoogd*/
            tl=tl+1;
/*de gap op de linkerrijstrook*/
            gapstr=p-voorgpass1;
            voorgpass1=p;
/*het gemiddelde van de gaps op de linkerrijstrook*/
            gemgap1=p/tl;
/*de som van de gekwadrateerde gaps op de linkerrijstrook*/
            somgapkwad1=somgapkwad1+(gapstr*gapstr);
            x=somgapkwad1/tl;
/*het kwadraat van het gemiddelde van de gaps op de
linkerrijstrook*/
            kwadgemgap=gemgap1*gemgap1;
/*de variantie en de variatiecoδfficiδnt op de
linkerrijstrook*/
            variantie=x-kwadgemgap;
            varcol=sqrt(variantie)/gemgap1;
/*uitvoer naar het bestand data*/
            fprintf(s,"%c%lf%d%d\n",strnr_p,p,sp*10,lengtel);
/*uitvoer naar een bestand*/
            if(u==1) fprintf(r,"%5ld %9.3lf %6.3lf
personen links   %6.3lf
%6.4lf\n",tr+tl,p,gapstr,gemgap1,varcol);
/*uitvoer naar het scherm*/
            fprintf(t,"%5ld %9.3lf %6.3lf         personen
links   %6.3lf
%6.4lf\n",tr+tl,p,gapstr,gemgap1,varcol);
/*uitvoer naar het bestand links*/
           fprintf(a,"%lf\n",gap_p);
/*als het programma tot een bepaald aantal voertuigen moet
lopen*/
          if (f==1)
              {
/*als het aantal voertuigen is bereikt*/
              if ((tr+tl)==aantal)
                  goto Stop;
              }
            }
/*als p NIET in het invloedsgebied ligt*/
         else
            {
/*p op de rechterrijstrook*/
            strnr_p='2';
/*het aantal voertuigen op de rechterrijstrook wordt met ΘΘn
verhoogd*/
            tr=tr+1;
/*de gap op de rechterrijstrook*/
            gapstr=p-voorgpass2;
            voorgpass2=p;
```

```
/*het gemiddelde van de gaps op de rechterrijstrook*/
            gemgap2=p/tr;
/*de som van de gekwadrateerde gaps op de rechterrijstrook*/
            somgapkwad2=somgapkwad2+(gapstr*gapstr);
            x=somgapkwad2/tr;
/*het kwadraat van het gemiddelde van de gaps op de
rechterrijstrook*/
            kwadgemgap=gemgap2*gemgap2;
/*de variantie en de variatiecoδfficiδnt op de
rechterrijstrook*/
            variantie=x-kwadgemgap;
            varco2=sqrt(variantie)/gemgap2;
/*uitvoer naar het bestand data*/
            fprintf(s,"%c%lf%d%d\n",strnr_p,p,sp*10,lengte1);
/*uitvoer naar een bestand*/
            if(u==1) fprintf(r,"%5ld %9.3lf        %6.3lf
personen rechts         %6.3lf
%6.4lf\n",tr+tl,p,gapstr,gemgap2,varco2);
/*uitvoer naar het scherm*/
            fprintf(t,"%5ld %9.3lf        %6.3lf  personen
rechts          %6.3lf
%6.4lf\n",tr+tl,p,gapstr,gemgap2,varco2);
/*uitvoer naar het bestand rechts*/
            fprintf(b,"%lf\n",gap_p);
/*als het programma tot een bepaald aantal voertuigen moet
lopen*/
            if (f==1)
                {
/*als het aantal voertuigen is bereikt*/
                if ((tr+tl)==aantal)
                    goto Stop;
                }
            }
/*genereer een toevalsgetal tussen 0 en 1*/
          do
            m=(double)rand()/RAND_MAX;
          while (m==0);
/*genereer gap en passagetijd van een personenauto*/
          gap_p=-log(m)/dp;
          p=p+gap_p;
          }
/*v op de rechterrijstrook*/
      strnr_v='2';
/*het aantal voertuigen op de rechterrijstrook wordt met ΘΘn
verhoogd*/
      tr=tr+1;
/*de gap op de rechterrijstrook*/
      gapstr=v-voorgpass2;
      voorgpass2=v;
/*het gemiddelde van de gaps op de rechterrijstrook*/
      gemgap2=v/tr;
/*de som van de gekwadrateerde gaps op de rechterrijstrook*/
      somgapkwad2=somgapkwad2+(gapstr*gapstr);
      x=somgapkwad2/tr;
/*het kwadraat van het gemiddelde van de gaps op de
rechterrijstrook*/
      kwadgemgap=gemgap2*gemgap2;
/*de variantie en de variatiecoδfficiδnt op de
```

```
rechterrijstrook*/
      variantie=x-kwadgemgap;
      varco2=sqrt(variantie)/gemgap2;
/*uitvoer naar het bestand data*/
      fprintf(s,"%c%lf%d%d\n",strnr_v,v,sv*10,lengte2);
/*uitvoer naar een bestand*/
      if(u==1) fprintf(r,"%5ld %9.3lf        %6.3lf  vracht
rechts          %6.3lf
%6.4lf\n",tr+tl,vh,gapstr,gemgap2,varco2);
/*uitvoer naar het scherm*/
      fprintf(t,"%5ld %9.3lf        %6.3lf  vracht   rechts
    %6.3lf          %6.4lf\n",tr+tl,vh,gapstr,gemgap2,varco2);
/*uitvoer naar het bestand rechts*/
      fprintf(b,"%lf\n",gap_v);
/*als het programma tot een bepaald aantal voertuigen moet
lopen*/
      if (f==1)
      {
/*als het aantal voertuigen is bereikt*/
          if ((tr+tl)==aantal)
            goto Stop;
          }
/*als er een toets word ingedrukt*/
    ch=0;
      if (kbhit())
          {
          getch();
/*onderbreek het programma totdat er weer een toets ingedrukt
wordt en
  lees de toets die ingedrukt is*/
          ch=getch();
          }
      } while (ch!=13);
  Stop:
  getch();
  clrscr();
/*uitvoer naar het scherm*/
  fprintf(t,"De verdeling van de gaps tussen de personenauto's
is exponentieel.\n");
  fprintf(t,"De verdeling van de gaps tussen de vrachtauto's
is ");
  if (verdlvr==1)
      fprintf(t,"regelmatig.\n\n");
  else
      fprintf(t,"exponentieel.\n\n");
  fprintf(t,"De dichtheid van de personenauto's :
%6.3lf\n",dp);
  fprintf(t,"De dichtheid van de vrachtauto's    :
%6.3lf\n\n",dv);
  fprintf(t,"De invloedslengte voor een personenauto   :
%6.3lf\n",Lv);
  fprintf(t,"De invloedslengte achter een personenauto :
%6.3lf\n\n",La);
  fprintf(t,"De snelheid van de personenauto's : %d\n",sp);
  fprintf(t,"De snelheid van de vrachtauto's   : %d\n\n",sv);
  fprintf(t,"De lengte van de personenauto's : %d\n",lengte1);
  fprintf(t,"De lengte van de vrachtauto's   :
%d\n\n",lengte2);
```

```
  fprintf(t,"Het aantal voertuigen op de linkerrijstrook  :
%ld\n",tl);
  fprintf(t,'Het aantal voertuigen op de rechterrijstrook :
%ld\n\n",tr);
  fprintf(t,"Gapgemiddelde op de linkerrijstrook  :
%6.3lf\n",gemgap1);
  fprintf(t,"Gapgemiddelde op de rechterrijstrook :
%6.3lf\n\n",gemgap2);
  fprintf(t,"De variatiecoδfficiδnt op de linkerrijstrook  :
%6.3lf\n",varco1);
  fprintf(t,"De variatiecoδfficiδnt op de rechterrijstrook :
%6.3lf\n",varco2);

/*uitvoer naar een bestand*/
  if (u==1)
  {
      fprintf(r,"\nDe verdeling van de gaps tussen de
personenauto's is exponentieel.\n");
      fprintf(r,"De verdeling van de gaps tussen de
vrachtauto's is ");
      if (verdlvr==1)
          fprintf(r,"regelmatig.\n\n");
    else
      fprintf(r,"exponentieel.\n\n");
      fprintf(r," \nDe dichtheid van de personenauto's :
%6.3lf\n\n",dp);
      fprintf(r,"De dichtheid van de vrachtauto's   :
%6.3lf\n",dv);
      fprintf(r,"De invloedslengte voor een personenauto   :
%6.3lf\n\n",Lv);
      fprintf(r,"De invloedslengte achter een personenauto :
%6.3lf\n",La);
      fprintf(r,"De snelheid van de personenauto's :
%d\n",sp);
      fprintf(r,"De snelheid van de vrachtauto's   :
%d\n\n",sv);
      fprintf(r,"De lengte van de personenauto's :
%d\n",lengte1);
      fprintf(r,"De lengte van de vrachtauto's   :
%d\n\n",lengte2);
      fprintf(r,"Het aantal voertuigen op de linkerrijstrook
: %ld\n",tl);
      fprintf(r,"Het aantal voertuigen op de rechterrijstrook
: %ld\n\n",tr);
      fprintf(r,"Gapgemiddelde op de linkerrijstrook  :
%6.3lf\n",gemgap1);
      fprintf(r,"Gapgemiddelde op de rechterrijstrook :
%6.3lf\n\n",gemgap2);
      fprintf(r,"De variatiecoδfficiδnt op de linkerrijstrook
: %6.3lf\n",varco1);
      fprintf(r,"De variatiecoδfficiδnt op de rechterrijstrook
: %6.3lf\n",varco2);
      }

/*de bestanden sluiten*/
  if (u==1 && r!=NULL)  fclose(r);
  if(s!=NULL) fclose(s);
  if(a!=NULL) fclose(a);
```

```
  if(b!=NULL) fclose(b);

  fprintf(t,"\nWilt u nogmaals het programma laten lopen?
(1=Ja  2=Nee) : ");
  scanf("%d",&g);

/*zolang het programma moeten worden herhaald*/
  } while (g==1);

  return 0;
}
```