



# basic education

Department:  
Basic Education  
**REPUBLIC OF SOUTH AFRICA**

**NASIONALE  
SENIOR SERTIFIKAAT**

**GRAAD 12**

**INFT.1**

**INLIGTINGSTEGNOLOGIE V1**

**NOVEMBER 2014**

**PUNTE: 150**

**TYD: 3 uur**

Hierdie vraestel bestaan uit 22 bladsye.

# OGGENDSESSIE

**INSTRUKSIES EN INLIGTING**

1. Hierdie vraestel is in DRIE afdelings verdeel. Kandidate moet AL DRIE afdelings beantwoord.
2. Die duur van hierdie eksamen is drie uur. As gevolg van die aard van hierdie eksamen is dit belangrik om kennis te neem dat jy nie toegelaat sal word om die eksamenlokaal voor die einde van die eksamensessie te verlaat nie.
3. Hierdie vraestel is opgestel met programmeringsterme wat nie programmeringstaal-spesifiek is nie (Delphi/Java (deur die Netbeans IDE te gebruik)).
4. Maak seker dat jy die vrae volgens die spesifikasies wat in elke vraag gegee word, beantwoord. Punte sal slegs volgens die voorgeskrewe spesifikasies toegeken word.
5. Beantwoord in elke vraag slegs wat gevra is. Byvoorbeeld, indien die vraag nie datavalidering vereis nie, sal geen punte vir datavalidering toegeken word nie.
6. Jou programme moet so gekodeer word dat dit met enige data sal werk en nie net met die voorbeelddata wat voorsien is of enige data-uittreksels wat in die vraestel verskyn nie.
7. Roetines soos soek, sorteer en seleksie moet vanuit eerste beginsels ontwikkel word. Jy mag nie die ingeboude funksies van 'n programmeringstaal vir enige van hierdie roetines gebruik nie.
8. Alle datastrukture moet deur jou as programmeerder verklaar word. Jy mag nie komponente wat in die koppelvlak voorsien is, gebruik om data te stoor en later weer op te roep nie.
9. Jy moet jou werk gereeld stoor op die disket wat aan jou gegee is of die skyfspasie wat aan jou toegeken is vir hierdie eksamensessie.
10. Maak seker dat jou eksamennummer as kommentaar in elke program verskyn wat jy kodeer, asook op elke gebeurtenis wat aangedui word.
11. Indien dit vereis word, druk die programmeringskode van al die programme/klasse wat jy voltooi het. Jy sal 'n halfuur tyd vir drukwerk gegun word na die eksamensessie.
12. Aan die einde van hierdie eksamensessie moet jy 'n disket/CD/DVD/geheuestokkie met al jou werk daarop gestoor, inlewer OF jy moet seker maak dat jou werk op die skyfspasie wat vir hierdie eksamensessie aan jou toegeken is, gestoor is. Maak seker dat alle lêers gelees kan word.

13. Die lêers wat jy benodig om hierdie vraestel te voltooi, is aan jou gegee op 'n disket/CD/DVD/geheuestokkie of op die skyfspasie wat aan jou toegeken is in die vorm van 'n wagwoordbeskermd, uitvoerbare lêer:

- Delphi-leerders moet die lêer **DelphiDataAFR.exe** gebruik
- Java-leerders moet die lêer **JavaDataAFR.exe** gebruik

Doen die volgende:

- Dubbelklik op die lêer
- Klik op die 'extract'-knoppie
- Sleutel die volgende wagwoord in: **Transport@(!\$**

**Lys van lêers wat in die lêergids DelphiDataAFR/JavaDataAFR voorsien is (nadat dit onttrek is):**

#### **Delphi-lêers**

##### **Vraag1:**

Vraag1\_P.dpr  
Vraag1\_P.res  
Vraag1\_U.dfm  
Vraag1\_U.pas

##### **Vraag2:**

Aflewering\_U.pas  
AfleweringInligting.txt  
Vraag2\_P.dpr  
Vraag2\_P.res  
Vraag2\_U.dfm  
Vraag2\_U.pas

##### **Vraag3:**

Vraag3\_P.dpr  
Vraag3\_P.res  
Vraag3\_U.dfm  
Vraag3\_U.pas

#### **Java (Netbeans)-lêers**

##### **Vraag1:**

Vraag1.form  
Vraag1.java

##### **Vraag2:**

Aflewering.java  
AfleweringInligting.txt  
Vraag2.form  
Vraag2.java

##### **Vraag3:**

Vraag3.form  
Vraag3.java

**SCENARIO:**

SuperTrans Afleweringdienste is 'n nasionale vervoermaatskappy met takke regoor Suid-Afrika. Jy word versoek om hulp te verleen met sommige van die programmatuurtoepassings wat die maatskappy beoog om binnekort te implementeer.

**AFDELING A****VRAAG 1: ALGEMENE PROGRAMMERINGSVAARDIGHEDE****INSTRUKSIES:**

Delphi-programmeerders	Java-programmeerders
<ul style="list-style-type: none"> <li>Die projek <b>Vraag1</b> is in die <b>DelphiDataAFR</b>-lêergids voorsien.</li> <li>Maak die onvoltooide projeklêer <b>Vraag1_P.dpr</b> in die <b>Vraag1</b>-lêergids oop.</li> <li>Voeg jou eksamennummer as kommentaar in die eerste reël van die hoofvormeenheid (<b>Vraag1_U</b>) -lêer by.</li> </ul>	<ul style="list-style-type: none"> <li>Die projek <b>Vraag1</b> is in die <b>JavaDataAFR</b>-lêergids voorsien.</li> <li>Maak die onvoltooide klas met die naam <b>Vraag1.java</b> wat in die lêergids <b>Source Packages (src)</b>, <b>Vraag1Package</b> voorkom, oop.</li> <li>Voeg jou eksamennummer as kommentaar in die eerste reël van die klas (<b>Vraag1</b>) by.</li> </ul>

**Doen die volgende:**

- Kompileer en voer die program uit. Die koppelvlak vertoon vyf verskillende afdelings met die opskrifte Vraag 1.1 tot Vraag 1.5. Die program het tans geen funksionaliteit nie. 'n Voorbeeld van die koppelvlak word hieronder gegee:

**Vraag 1.1**

Selekteer die vertrekpunt: Kaapstad

Selekteer die bestemming: Kaapstad

Sleutel die getal kilometer in:

Bevestig aflerwing: Aflerwingsetiket

**Vraag 1.2**

Selekteer die kategorie:

- A1 (0kg - 1kg)
- A2 (>1kg - 2kg)
- A3 (>2kg - 5kg)
- A4 (>5kg)

Merk as spoedpos vereis word:

☐ Spoedpos

Afleringskoste:

**Vraag 1.3**

Afleringsboks se nommer:

**Vraag 1.4**

UPK-strepieskode: 639382000393 Valideer strepieskode

**Vraag 1.5**

Selekteer 'n stad: Kaapstad Vertoon en stoor aflerings

- Voltooi die kode vir elke afdeling van VRAAG 1 soos in VRAAG 1.1 tot VRAAG 1.5 hieronder beskryf word.

### 1.1 Knoppie – [Bevestig aflewering]

Verkry die volgende data uit die relevante komponente:

- Vertrekpunt uit die **Vertrekpunt**-kombinasielys ('combo box')
- Bestemming uit die **Bestemming**-kombinasielys ('combo box')
- Getal kilometer uit die **Kilometer**-teksblokkie ('text box')

Skep 'n reël met teks as afvoer wat die vertrekpunt, die bestemming en die getal kilometer aandui, soos in die voorbeeld hieronder getoon word. Plaas die saamgestelde reël teks in die etiketkomponent ('label component') wat voorsien is.

Voorbeeld van moontlike toevoer:

Selekteer die vertrekpunt	Johannesburg ▼
Selekteer die bestemming	Durban ▼
Sleutel die getal kilometer in	635

Vereiste afvoer:

Afleweringsetiket
Johannesburg na Durban : 635 km

(5)

### 1.2 Knoppie – [Afleweringstkoste]

Die volgende komponente word voorsien:

- 'n Lysblokkie ('list box') wat die gewigskategorieë van aflewering in terme van kodes (A1–A4) in die volgende formaat aandui:

<kode><spasie><(interval in kg)>

A1 (0kg - 1kg)
A2 (>1kg - 2kg)
A3 (>2kg - 5kg)
A4 (>5kg)

Die volgende tarief per gewigskategorie is van toepassing:

Kategorie	Tarief per kilometer
A1	R0,60
A2	R1,00
A3	R1,25
A4	R1,65

- 'n Merkblokkie ('check box') wat aandui of spoedpos gebruik moet word.  
'n Standaardbedrag van R100,00 word vir spoedpos gehef.

Wanneer die gebruiker op die **Afleweringenskoste**-knoppie klik, moet die getal kilometer wat in VRAAG 1.1 ingesleutel is, die gewigskategorie wat gekies is en of spoedpos vereis word of nie, gebruik word om die afleweringenskoste te bereken.

Voorbeeld van die afvoer indien die getal kilometer 635 is, die gewig van die item wat afgelewer word in die A2-kategorie is, en spoedpos vereis word:



(10)

### 1.3 Knoppie – [Afleweringboks se nommer]

Items wat afgelewer moet word, moet in spesifieke afleweringbokse geplaas word. Die regte afleweringboks vir elke individuele item moet bepaal word deur die kriteria hieronder te gebruik:

- Daar is vyf afleweringbokse wat van 1 tot 5 genommer is.
- Alle **spoedpositems** moet in afleweringboks 4 geplaas word.
- Alle ander items wat afgelewer moet word, sal **ewekansig** ('randomly') in die oorblywende afleweringbokse (1, 2, 3 of 5) geplaas word.

Vertoon die nommer van die afleweringboks waarin die item geplaas moet word.

Voorbeeld van afvoer indien spoedpos in VRAAG 1.2 versoek is (op die volgende bladsy):

**Vraag 1.2**

Selekteer die kategorie

A1 (0kg - 1kg)  
**A2 (>1kg - 2kg)**  
 A3 (>2kg - 5kg)  
 A4 (>5kg)

Merk as spoedpos vereis word

☒ Spoedpos

Afleweringkoste R735.00

**Vraag 1.3**

Afleweringboks se nommer 4

Voorbeeld van afvoer indien spoedpos NIE in VRAAG 1.2 vereis word NIE:

**Vraag 1.2**

Selekteer die kategorie

A1 (0kg - 1kg)  
**A2 (>1kg - 2kg)**  
 A3 (>2kg - 5kg)  
 A4 (>5kg)

Merk as spoedpos vereis word

☐ Spoedpos

Afleweringkoste R635.00

**Vraag 1.3**

Afleweringboks se nommer 3

**LET WEL:** As gevolg van die aard van die ewekansige ('random') funksie mag die waarde van die afleweringboks wat in die skermkoot hierbo vertoon word, verskil van die waarde wat deur jou program vertoon word.

(9)

#### 1.4 Knoppie – [Valideer strepieskode]

'n Universele Produkkode (UPK) ('Universal Product Code (UPC)') -strepieskode word op items wat afgelewer moet word, gedruk. Die prentjie op die volgende bladsy toon 'n voorbeeld van 'n UPK-strepieskode:



Die nommer van die strepieskode bestaan uit twaalf syfers. Byvoorbeeld, die nommer van die strepieskode wat in die prentjie hierbo getoon word, is 639382000393. Die laaste syfer van 'n UPK word 'n kontrolesyfer genoem. Die kontrolesyfer word deur 'n skandeerder gebruik om te bepaal of 'n strepieskode geldig is of nie.

Gebruik die algoritme hieronder en skryf kode vir die **Valideer strepieskode**-knoppie. Die kode moet die kontrolesyfer kontroleer en 'n boodskap vertoon wat aandui of die strepieskode geldig is of nie.

#### Algoritme:

1. strepieskode  $\leftarrow$  teksblokkie se waarde
2. somOnewePlekke  $\leftarrow 0$
3. somEwePlekke  $\leftarrow 0$
4. lus van die eerste syfer tot by die tweede laaste syfer van strepieskode
5. indien die logikaposisie van die syfer in die strepieskode ewe is  
     somEwePlekke  $\leftarrow$  somEwePlekke + syfer by posisie  
   anders  
     somOnewePlekke  $\leftarrow$  somOnewePlekke + syfer by posisie
6. som  $\leftarrow$  somOnewePlekke \* 3 + somEwePlekke
7. kontrolesyfer  $\leftarrow 10 - (\text{som modulus } 10)$
8. indien kontrolesyfer = laaste syfer van strepieskode  
     vertoon die kontrolesyfer en 'n gepaste boodskap wat aandui dat die strepieskode geldig is  
   anders  
     vertoon 'n gepaste boodskap wat aandui dat die strepieskode ongeldig is

Voorbeeld:

Indien die strepieskode 639382000393 is, dan is:

$$\text{somOnewePlekke} = 6+9+8+0+0+9 = 32$$

$$\text{somEwePlekke} = 3+3+2+0+3 = 11$$

Voorbeeld van moontlike afvoer vir 'n geldige strepieskode:

UPK-strepiesskode	639382000393	<div style="border: 1px solid black; padding: 2px 10px; display: inline-block;">Valideer strepiesskode</div>
<div style="border: 1px solid black; padding: 5px;">Die strepiesskode is geldig. Kontrolesyfer: 3</div>		



Voorbeeld van moontlike afvoer vir 'n ongeldige strepieskode:

The screenshot shows a web form with a label 'UPK-strepietoonkode' and a text input field containing '535682000398'. To the right of the input field is a button labeled 'Valideer strepietoonkode'. Below the input field, a message box displays the text: 'Die strepietoonkode is NIE geldig nie. Korrekte kontrolesyfer: 5'.

(14)

### 1.5 Knoppie – [Vertoon en stoor aflewerings]

Al die aflewerings vir Desember 2014 is in die gegewe skikking met die naam **arrDesAflewerings** gestoor. Die formaat van elke inskrywing in die skikking is soos volg:

<datum><spasie><vertrekpunt><spasie>na<spasie><bestemming>

Voorbeeld:

2014-12-01 Durban na Kaapstad

Die gebruiker moet 'n stad uit die kombinasie wat voorsien is, kies. Al die aflewerings gedurende Desember 2014 na of van die stad wat gekies is, moet in die afvoerarea wat voorsien is, vertoon word en ook na 'n tekslêer geskryf word.

Skryf kode om die volgende te doen:

- Skep 'n tekslêer waar die naam van die lêer saamgestel is deur die teks 'Desember2014' met die naam van die stad wat gekies is, te kombineer.

Voorbeeld:

Indien Durban gekies is, moet die naam van die tekslêer **Desember2014Durban.txt** wees.

- Gebruik die data wat in die **arrDesAflewerings**-skikking gestoor is en vertoon die aflewerings na en van die stad wat gekies is in die afvoerarea wat voorsien is. Gebruik die naam van die stad as 'n opskrif.
- Stoor die aflewerings na en van die stad wat gekies is in die tekslêer wat geskep is – een aflewering per reël.

Voorbeeld van die inhoud van die afvoerarea indien die stad Durban gekies word:

**Vraag 1.5**

Selekteer 'n stad

Durban  
2014-12-01 Durban na Kaapstad  
2014-12-04 Port Elizabeth na Durban  
2014-12-05 Durban na Kimberley  
2014-12-05 Durban na Potchefstroom  
2014-12-07 Potchefstroom na Durban  
2014-12-09 Durban na Bloemfontein

Voorbeeld van die inhoud van die **Desember2014Durban.txt**-tekslêer:

```
2014-12-01 Durban na Kaapstad
2014-12-04 Port Elizabeth na Durban
2014-12-05 Durban na Kimberley
2014-12-05 Durban na Potchefstroom
2014-12-07 Potchefstroom na Durban
2014-12-09 Durban na Bloemfontein
```

(12)

- Sleutel jou eksamennummer as kommentaar in die eerste reël van die programlêer in.
- Stoor jou program.
- 'n Drukstuk van die kode mag vereis word.

**TOTAAL AFDELING A: 50**

**AFDELING B****VRAAG 2: OBJEK-GEORIËNTEERDE PROGRAMMERING**

SuperTrans Afleweringsdienste besit vyf vragmotors. Die vragmotors word as lig-, medium- of swaardiensvragmotors geklassifiseer. Die vragmotors word op vier verskillende roetes (RN1 tot RN4) gebruik om vragte af te lewer.

**INSTRUKSIES:**

<b>Delphi-programmeerders</b>	<b>Java-programmeerders</b>
<p>Die projek <b>Vraag2</b> is in die <b>DelphiDataAFR</b>-lêergids voorsien en bevat:</p> <ul style="list-style-type: none"> <li>○ 'n Hoofvormeenheid met die naam <b>Vraag2_U.pas</b></li> <li>○ 'n Onvoltooide eenheidlêer met die naam <b>Aflewering_U.pas</b></li> <li>○ 'n Tekslêer (<b>AfleweringInligting.txt</b>) wat inligting oor aflewerings bevat</li> </ul> <ul style="list-style-type: none"> <li>• Maak die onvoltooide projeklêer met die naam <b>Vraag2_P.dpr</b> in die <b>Vraag2</b>-lêergids oop.</li> <li>• Besigtig ('View') (Ctrl+F12) die eenheidlêer <b>Aflewering_U.pas</b> en voeg jou eksamennummer as kommentaar in die eerste reël van beide lêer <b>Vraag2_U.pas</b> en <b>Aflewering_U.pas</b> in.</li> </ul>	<p>Die projek <b>Vraag2</b> is in die <b>JavaDataAFR</b>-lêergids voorsien en bevat:</p> <ul style="list-style-type: none"> <li>○ 'n GGK ('GUI') -klaslêer met die naam <b>Vraag2.java</b></li> <li>○ 'n Onvoltooide objekklaslêer met die naam <b>Aflewering.java</b></li> <li>○ 'n Tekslêer (<b>AfleweringInligting.txt</b>) wat inligting oor aflewerings bevat</li> </ul> <ul style="list-style-type: none"> <li>• Maak die onvoltooide klas met die naam <b>Vraag2.java</b> en <b>Aflewering.java</b> in die lêergids <b>Source Packages (src)</b>, <b>Vraag2Package</b> oop.</li> <li>• Voeg jou eksamennummer as kommentaar in die eerste reël van beide klas <b>Vraag2.java</b> en <b>Aflewering.java</b> in.</li> </ul>

Doen die volgende:

- Kompileer en voer die program uit. Die program het tans geen funksionaliteit nie. 'n Voorbeeld van die koppelvlak verskyn op die volgende bladsy:

SuperTrans Afleweringdienste	
<b>Skep en vertoon nuwe aflewering-objek</b> Kies vragmotor se nommer <input type="text" value="Tr1"/> <input type="button" value="Kry data uit lêer"/> Nuwe aflewering nommer <input type="text"/> Begin-odometerlesing <input type="text"/> Sleutel eind-odometerlesing in <input type="text"/> <input type="button" value="Nuwe aflewering"/> <input type="button" value="Vertoon aflewering"/> <div style="border: 1px solid black; height: 100px; width: 100%;"></div>	<b>Brandstof gebruik</b> Werklike brandstof gebruik <input type="text"/> <input type="button" value="Kontroleer brandstof gebruik"/> <input type="text"/>  <b>Tolgeld</b> Roete om te volg <input type="text"/> <input type="button" value="Bereken tolgeld"/> <input type="text"/>

- Voltooi die kode vir hierdie program soos in VRAAG 2.1 en VRAAG 2.2 hieronder gespesifiseer word.

2.1 Die gegewe onvoltooide objekklas (**TAflewering/Aflewering**) bevat die volgende kode:

- Die verklaring van vyf attribute wat 'n **aflewering**-objek beskryf
- Die verklaring van 'n tweedimensionele skikking wat gebruik moet word om tolgeld te bepaal
- 'n **toString**-metode

Die attribute van 'n **aflewering**-objek is die volgende:

Name van attribute		
Delphi	Java	Beskrywing
fAfweringNom	afleweringNom	Nommer wat aan 'n spesifieke aflewering toegeken is, byvoorbeeld 1, 2, ensovoorts
fVragmotorNom	vragmotorNom	Vragmotor se nommer, byvoorbeeld Tr1, Tr2, Tr3, Tr4 of Tr5
fBrandstofGebruik	brandstofGebruik	Brandstof wat tydens die aflewering gebruik is
fOdoBegin	odoBegin	Odometerlesing aan die begin van die aflewering
fOdoEind	odoEind	Odometerlesing na voltooiing van die aflewering

Voltooi die kode in die gegewe **aflewering**-klas (**TAflewering/Aflewering**) soos in VRAAG 2.1.1 tot VRAAG 2.1.4 hieronder beskryf word:

- 2.1.1 Skryf kode vir 'n konstruktormetode wat die aflewering se nommer, vragmotor se nommer, odometerlesing aan die begin van die aflewering en die odometerlesing na voltooiing van die aflewering as parameterwaardes ontvang. Ken hierdie waardes aan die relevante attribute van die objekklas toe. (3)
- 2.1.2 Skryf 'n wysigingsmetode ('mutator method') en 'n toegangsmetode ('accessor method') vir die **fBrandstofGebruik/brandstofGebruik**-attribuut. (4)
- 2.1.3 Skryf 'n metode met die naam **berekenAfstand** om die afstand wat afgelê is, te bereken en terug te stuur volgens die begin- en eind-odometerlesings van die aflewering. (3)
- 2.1.4 Verskillende tolgelde moet op verskillende tolroetes betaal word. Die roetes wat gebruik word, is RN1, RN2, RN3 en RN4. Tolgeld op hierdie roetes is afhanklik van die tipe vragmotor wat gebruik word. Die maatskappy se vragmotors is soos volg geklassifiseer:

- Ligtediensvragmotors: Tr1, Tr2
- Mediumdiensvragmotor: Tr3
- Swaardiensvragmotors: Tr4, Tr5

'n Tweedimensionele skikking met die naam **tolGeld** bevat die tolgeld vir die verskillende roetes vir verskillende tipes vragmotors en is as deel van die gegewe kode voorsien. Die inhoud van die skikking kan soos volg voorgestel word:

	Ligtediens- vragmotor	Mediumdiens- vragmotor	Swaardiens- vragmotor
RN1	R105,50	R135,00	R210,00
RN2	R35,00	R54,00	R82,00
RN3	R85,00	R129,00	R205,00
RN4	R112,00	R170,00	R219,00

Rye:               Verteenwoordig die roetes RN1 tot RN4  
Kolomme:       Verteenwoordig die tipes vragmotors

Skryf 'n metode met die naam **bepaalTolGeld** om die tolgeld wat vir die aflewering betaalbaar is, te bepaal en terug te stuur. Die metode moet die roete (RN1, RN2, RN3 of RN4) as 'n parameter ontvang. Gebruik die tweedimensionele skikking met die naam **tolGeld** om die tolgeld vir die roete en tipe vragmotor wat vir die aflewering gebruik is, op te soek.

**LET WEL:** Dit is verpligtend om die gegewe tweedimensionele skikking in jou oplossing te gebruik om die tolgeld op te soek. (10)

- 2.2 'n Tekslêer met die naam **AfleveringsInligting.txt** bevat 'n onbekende getal reëls met inligting oor vorige voltooide afleverings. Elke reël met inligting bevat data oor 'n enkele aflevering in die volgende formaat:

**<afleveringsnommer>#<vragmotor se nommer>#<odometerlesing na voltooiing van die aflevering>**

Voorbeeld van sommige van die data in die tekslêer met die naam **AfleveringsInligting.txt**:

```
1#Tr1#121110
2#Tr2#8010
3#Tr3#15021
4#Tr4#700
5#Tr1#121453
6#Tr3#15653
:
```

Die data van die eerste aflevering kan soos volg geïnterpreteer word:

- Aflevering **1** identifiseer die aflevering.
- Vragmotor **Tr1** is vir die aflevering gebruik.
- Die lesing op die odometer na voltooiing van die aflevering was **121110**.

Doen die volgende om die kode vir elke knoppie in die hoofvormeenheid (Delphi)/GGK ('GUI')-klas (Java) soos hieronder beskryf word, te voltooi:

### 2.2.1 Knoppie – [Kry data uit lêer]

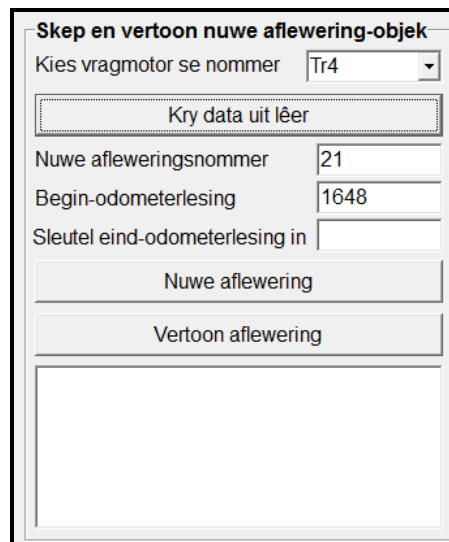
- Kies die spesifieke vragmotor wat gebruik gaan word uit die gegewe **Vragmotornommer**-kombinasielys.
- Gebruik die gegewe tekslêer **AfleveringsInligting.txt** om die volgende te bepaal:
  - **Nuwe afleveringsnommer**  
Die afleveringsnommer vir die nuwe aflevering sal op die laaste aflevering wat in die tekslêer gestoor is, se nommer volg. Indien die inligting van 20 afleverings in die tekslêer gestoor is, sal die nommer van die nuwe aflevering 21 wees.
  - **Begin-odometerlesing**  
Die odometerlesing van die laaste aflevering wat deur die geselekteerde vragmotor gedoen is, moet as die begin-odometerlesing vir die nuwe aflevering gebruik word.

Voorbeeld:

Indien vragmotor Tr4 gekies is en die odometerlesing wat na voltooiing van die laaste aflevering vir Tr4 in die tekslêer gestoor is 1648 is, moet die waarde 1648 aan die begin van die nuwe aflevering as die odometerlesing gebruik word.

- Vertoon die **nuwe afleweringsnommer** en die **begin-odometerlesing** in die teksblokkies wat voorsien is.
- Vertoon 'n gepaste boodskap in 'n dialoogblokkie en beëindig die program indien geen toegang tot die tekslêer verkry kan word nie.

Voorbeeld van die inhoud van die relevante teksblokkies indien vragmotor Tr4 gekies is:



**LET WEL:** Indien jy nie die vereiste inligting uit die gegewe tekslêer kan lees nie, sleutel die nommer en begin-odometerlesing van die nuwe aflewering in die teksblokkies in om met die res van die program te kan voortgaan.

(14)

### 2.2.2 Knoppie – [Nuwe aflewering]

Om 'n nuwe **Aflewering**-objek te skep moet die gebruiker eers die eind-odometerlesing vir die aflewering in die gegewe teksblokkie insleutel.

**LET WEL:** Die eind-odometerlesing wat die gebruiker insleutel moet groter wees as die vorige odometerlesing vir die vragmotor, omdat dit die begin-odometerlesing vir hierdie aflewering is. Geen validering is nodig nie.

**LET WEL:** Die **Aflewering**-objek se veranderlike is reeds globaal verklaar as deel van die gegewe kode.

Skryf kode vir die **Nuwe aflewering**-knoppie om die volgende te doen:

- Gebruik die nuwe afleweringsnommer, die vragmotor se nommer, die odometerlesing aan die begin en die odometerlesing aan die einde van die aflewering om die **Aflewering**-objek te skep.
- Vertoon 'n boodskap wat aandui dat die objek suksesvol geskep is.

Aflewering-objek is suksesvol geskep.

- Kry die afstand wat afgelê is deur die **berekenAfstand**-metode te roep en die geskatte liter brandstof wat vir die aflewering gebruik is, te bereken. Gebruik die volgende inligting:

Een liter brandstof word gebruik vir elke vyf kilometer wat afgelê word.

Gebruik die metode wat jy in VRAAG 2.1 geskryf het om die **fBrandstofGebruik**-/brandstofGebruik-attriboot gelyk te stel aan hierdie berekende waarde.

**LET WEL:** Die volgende knoppies moet geaktiveer word:

- Kontroleer brandstof gebruik
- Bereken tolgeld

(10)

### 2.2.3 Knoppie – [Vertoon aflewering]

Vertoon die objek in die afvoerarea wat voorsien is deur die **toString**-metode te gebruik.

Voorbeeld van die afvoer vir vragmotor Tr4 met 'n eind-odometerlesing van 2000:

Vertoon aflewering

Afleveringsnommer: 21  
Vragmotornommer: Tr4  
Odometerlesing:  
    (Begin) 1648  
    (Einde) 2000  
Brandstof gebruik: 70.4 liter

(2)



## 2.2.4 Knoppie – [Kontroleer brandstof gebruik]

Daar word van die bestuurder van die vragmotor verwag om die brandstof tenk na afloop van elke aflewering te hervul. Weens die aard van die roetes, mag die hoeveelheid brandstof wat nodig is om die tenk te hervul soms van die hoeveelheid brandstof wat vir die aflewering bereken is, verskil.

Skryf kode om die volgende te doen:

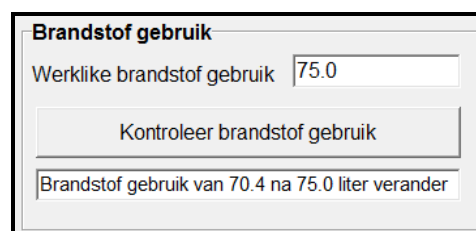
- Sleutel die hoeveelheid brandstof wat gebruik word om die tenk te hervul in die gegewe teksblokkie in.

**WENK:** Die waarde 75 kan gebruik word om die kode te toets.

- Kry die berekende hoeveelheid brandstof wat gebruik is uit die relevante attribuut in die objekklas.
- Bereken die verskil tussen die brandstof wat gebruik is om die tenk te hervul en die berekende hoeveelheid brandstof wat in die relevante attribuut van die **Aflewering**-objek gestoor is.
  - Indien die verskil minder as 10% is, moet die **fBrandstofGebruik-/brandstofGebruik**-attribuut verander word deur die waarde gelyk te stel aan die nuwe waarde wat in die teksblokkie ingesleutel is.

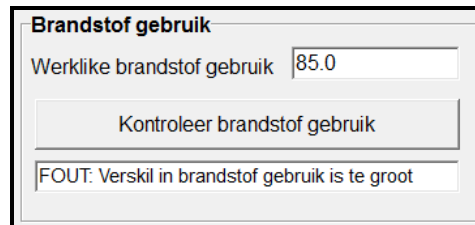
'n Gepaste boodskap moet in die gegewe teksblokkie vertoon word om aan te dui of die waarde van die **fBrandstofGebruik-/brandstofGebruik**-attribuut verander is of nie.

Voorbeeld van die afvoer indien die aflewering deur vragmotor Tr4 gedoen is en die werklike brandstof gebruik 75,0 is:



- Indien die verskil groter of gelyk aan 10% is, moet 'n foutboodskap vertoon word en die waarde van die **fBrandstofGebruik-/brandstofGebruik**-attribuut moet nie verander word nie.

Voorbeeld van die afvoer indien die aflewering deur vragmotor Tr4 gedoen is en die werklike brandstof gebruik 85,0 is:



(9)

### 2.2.5 Knoppie – [Bereken tolgeld]

Om die tolgeld te bepaal vir die roete wat vir die nuwe aflewering gebruik word, moet die roete se nommer (byvoorbeeld RN3) in die gegewe teksblokkie ingesleutel word. Hierdie roetenommer moet as 'n parameter na die **bepaalTolGeld**-metode gestuur word.

Die tolgeld moet in die gegewe etiketkomponent vertoon word. Die bedrag moet as geldeenheid (rand) tot TWEE desimale plekke vertoon word.

Voorbeeld van die afvoer indien die aflewering deur vragmotor Tr4 op roete RN3 gedoen is:



(5)

- Maak seker dat jou eksamenommer as kommentaar in die eerste reël van die klas en die vorm verskyn.
- Stoor al die lêers.
- Druk die kode vir beide klasse wat jy geskep het, indien drukstukke verlang word.

**TOTAAL AFDELING B: 60**

**AFDELING C****VRAAG 3: PROBLEEMOPLOSSING**

SuperTrans Afleweringsdienste bied 'n daaglikse spoeddiens van Kaapstad na Johannesburg aan. Die stoorruimte in die vragmotor wat vir die aflewerings gebruik word, is in twee rakke vir breekbare en nie-breekbare items onderskeidelik verdeel. Die maksimum kapasiteit van die rak wat vir breekbare items gereserveer is, is 20 items terwyl 'n maksimum van 30 items op die rak vir nie-breekbare items geplaas kan word. Die laaisone by die maatskappy beheer die verwydering van 'n vrag, die laai van die items in die trok en die kontrolering van die status van die vrag in die trok op enige gegewe tydstop.

'n Onvoltooide program om die laai van die vragmotors te bestuur, is voorsien.

**INSTRUKSIES:**

<b>Delphi-programmeerders</b>	<b>Java-programmeerders</b>
<ul style="list-style-type: none"> <li>Die projek <b>Vraag3</b> is in die <b>DelphiDataAFR</b>-lêergids voorsien.</li> <li>Maak die onvoltooide projeklêer <b>Vraag3_P.dpr</b> in die <b>Vraag3</b>-lêergids oop.</li> <li>Voeg jou eksamennummer as kommentaar in die eerste reël van die hoofvormeenheid (<b>Vraag3_U</b>)-lêers by.</li> </ul>	<ul style="list-style-type: none"> <li>Die projek <b>Vraag3</b> is in die <b>JavaDataAFR</b>-lêergids voorsien.</li> <li>Maak die onvoltooide klas met die naam <b>Vraag3.java</b> in die lêergids <b>Source Packages (src)</b>, <b>Vraag3Package</b> oop.</li> <li>Voeg jou eksamennummer as kommentaar in die eerste reël van die klas (<b>Vraag3</b>) by.</li> </ul>

Doen die volgende:

- Kompileer en voer die program uit. Die program het tans geen funksionaliteit nie. 'n Voorbeeld van die koppelvlak word hieronder gegee.

Die afvoerarea van die GGK ('GUI') met die naam 'Laaivordering-vertoonarea' verteenwoordig die stoorarea van die vragmotor. 'n Sterretjie (\*) word gebruik om 'n item wat op 'n rak geplaas is, voor te stel.

Voorbeeld van die afvoer van 'n volgelaai vragmotor met 20 breekbare en 30 nie-breekbare items:

```
Laaivordering-vertoonarea:
=====
Breekbare items:          *****
Nie-breekbare items:     *****
```

- Skryf kode om VRAAG 3.1 tot VRAAG 3.3 soos in die instruksies hieronder verduidelik word, te voltooi:

### 3.1 Knoppie – [Laai item]

Skryf kode om die volgende te doen wanneer 'n item gelaai word:

- Skep 'n laaikode.
- Voeg die item, wat deur 'n sterretjie (\*) voorgestel word, op die regte rak in die laaivordering-vertoonarea by.

#### Laaikode:

Die laaikode word saamgestel uit die letter 'B' vir breekbare items en die letters 'NB' vir nie-breekbare items, gevolg deur die reeksnommer van die item op die rak.

Byvoorbeeld:

B1 verwys na item 1 op die rak vir breekbare items.

NB6 verwys na item 6 op die rak vir nie-breekbare items.

#### Voeg item by laaivordering-vertoonarea:

Indien daar plek op die rak is, skep en vertoon die laaikode in die teksblokkie wat voorsien word en dateer die laaivordering-vertoonarea op om die nuwe item te vertoon.

Indien die item nie gelaai kan word nie, moet die laaikode oopgelaat word en 'n dialoogblokkie moet gebruik word om die volgende boodskap te vertoon:  
***'Laai van item kan nie verwerk word nie – Geen laaispasie nie'***

Voorbeeld van afvoer van die 'Laaivordering-vertoonarea' indien die item wat gelaai word 'n nie-breekbare item is. In hierdie voorbeeld is vyf breekbare items reeds gelaai (op die volgende bladsy):

Voorbeeld van 'n poging om 'n breekbare item te laai as die rak vir breekbare items reeds vol is:

(20)

3.2 **LET WEL:** Indien jy nie VRAAG 3.1 kon voltooi nie, gebruik die data hieronder om VRAAG 3.2 te voltooi:

Getal breekbare items: 4  
Getal nie-breekbare items: 13

### Knoppie – [Kontroleer vraagstatus]

'n Aflewering kan gedoen word indien die vragmotor 'n minimum vraag het van:

- 50% breekbare items (10 breekbare items), en
- 50% nie-breekbare items (15 nie-breekbare items)

Wanneer hierdie knoppie geklik word, moet 'n vraagstatusverslag 'n opsomming van beide die getal breekbare en nie-breekbare items vertoon, deur die volgende kolomopskrifte te gebruik:

Tipe item	Getal items	Persentasie gelaai
-----------	-------------	--------------------

- Indien die persentasie wat gelaai is, 50% of meer vir beide breekbare en nie-breekbare items is, moet die boodskap '**Die aflewering mag voortgaan**' vertoon word.

- Indien die persentasie items wat gelaai is minder as 50% vir breekbare OF nie-breekbare items is, moet die boodskap '**Die aflewering mag nie voortgaan nie**' vertoon word, asook die getal uitstaande items van elke tipe wat nodig is om die minimum vraag te bereik.

**LET WEL:** Punte sal vir kolomformatering en die vertoon van die persentasies tot TWEE desimale plekke toegeken word.

Voorbeeld 1: Indien die vraagstatus van die vragmotor 4 breekbare items en 10 nie-breekbare items is, moet die vraagstatusverslag soos volg lyk:

```
Vraagstatusverslag:
=====
Item tipe      Getal items      Persentasie gelaai
Breekbaar      4                20.00
Nie-breekbaar 10                33.33

Die aflewering mag nie voortgaan nie.
Getal breekbare items nog benodig: 6
Getal nie-breekbare items nog benodig: 5
```

Voorbeeld 2: Indien die vraagstatus van die vragmotor 12 breekbare items en 17 nie-breekbare items is, moet die vraagstatusverslag soos volg lyk:

```
Vraagstatusverslag:
=====
Item tipe      Getal items      Persentasie gelaai
Breekbaar      12               60.00
Nie-breekbaar 17               56.67

Die aflewering mag voortgaan.
```

(17)

### 3.3 Knoppie – [Verwyder vraag]

Inisialiseer al die veranderlikes en datastrukture om vir 'n nuwe vraag voor te berei. Verwyder ook alle teks uit die afvoerarea.

(3)

- Sleutel jou eksamennummer as kommentaar in die eerste reël van die programlêer in.
- Stoor jou program.
- 'n Drukstuk van die kode mag vereis word.

**TOTAAL AFDELING C: 40**  
**GROOTTOTAAL: 150**



- If the percentage of items loaded is less than 50% for fragile OR non-fragile items, then the message **'The delivery may not progress'** must be displayed along with the outstanding number of items for each type required to make up the minimum load.

**NOTE:** Marks will be awarded for column formatting and for displaying the percentage loaded to TWO decimal places.

Example 1: If the load status of the truck is 4 fragile items and 10 non-fragile items, the load status report should be as follows:

Load status report:			
=====			
Item type	Number of items	Percentage loaded	
Fragile	4	20.00	
Non-fragile	10	33.33	
The delivery may not progress.			
Number of fragile items still required: 6			
Number of non-fragile items still required: 5			

Example 2: If the load status of the truck is 12 fragile items and 17 non-fragile items, the load status report should be as follows:

Load status report:			
=====			
Item type	Number of items	Percentage loaded	
Fragile	12	60.00	
Non-fragile	17	56.67	
The delivery may progress.			

3.3

**Button – [Clear load]**

Initialise all variables and data structures to prepare for a new load. Also clear all text from the output area.

(3)

- Enter your examination number as a comment in the first line of the program file.
- Save your program.
- A printout of the code may be required.

**TOTAL SECTION C:**  
**GRAND TOTAL:**

40  
150



Example of an attempt to load a fragile item when the shelf for fragile items is full:

(20)

3.2

**NOTE:** If you could not complete QUESTION 3.1, use the data below to complete QUESTION 3.2:

Number of fragile items: 4  
Number of non-fragile items: 13

### Button – [Check load status]

The delivery can be made if the truck has a minimum load of:

- 50% fragile items (10 fragile items), and
- 50% non-fragile items (15 non-fragile items)

When this button is clicked, a load status report must display a summary of the number of both fragile and non-fragile items, using the following column headings:

Item type	Number of items	Percentage loaded
-----------	-----------------	-------------------

- If the percentage of items loaded is 50% or more for both fragile and non-fragile items, the message 'The delivery may progress' must be displayed.





The output area of the GUI labelled 'Loading progress display area' represents the storage area of the truck. An asterisk (\*) is used to represent an item placed on a shelf.

Example of the display of a fully loaded truck with 20 fragile and 30 non-fragile items:

```

Loading progress display area:
=====
Fragile items: *****
Non-fragile items: *****

```

- Write code to complete QUESTION 3.1 to QUESTION 3.3 as explained in the instructions below:

### 3.1 Button – [Load item]

Write code to do the following when an item is loaded:

- Create a loading code.
- Add the item, represented by an asterisk (\*), to the correct shelf in the loading progress display area.

#### Loading code:

The loading code is compiled using the letter 'F' for fragile items and the letters 'NF' for non-fragile items, followed by the sequence number of the item on the shelf.

For example:

F1 refers to item 1 on the shelf for fragile items.  
NF6 refers to item 6 on the shelf for non-fragile items.

#### Add item to loading progress display area:

If there is space on the shelf, create and display the loading code in the text box provided and update the loading progress display area to show the new item.

If the item cannot be loaded, the loading code must be left empty and a dialog box must be used to display the following message: **'Loading of item cannot be processed – No loading space'**

Example of output of the 'Loading progress display area' if the item loaded is the first non-fragile item. In this example five fragile items have already been loaded (on the next page):



## SECTION C

### QUESTION 3: PROBLEM-SOLVING

SuperTrans Courier Services provides a daily speed service from Cape Town to Johannesburg. The storage space in the truck used for the deliveries is divided into two shelves for fragile and non-fragile items respectively. The maximum capacity of the shelf reserved for fragile items is 20 items while a maximum of 30 items can be placed on the shelf reserved for non-fragile items. The loading zone at the company controls the clearing of a load, loading of items into the truck and checking the status of the truck load at any given time.

An incomplete program has been provided to manage the loading of trucks.

#### INSTRUCTIONS:

Delphi programmers	Java programmers
<ul style="list-style-type: none"> <li>The project <b>Question3</b> is provided in the <b>DelphiDataENG</b> folder.</li> <li>Open the incomplete project file <b>Question3_P.dpr</b> in the <b>Question3</b> folder.</li> <li>Add your examination number as a comment in the first line of the main form unit (<b>Question3_U</b>) files.</li> </ul>	<ul style="list-style-type: none"> <li>The project <b>Question3</b> is provided in the <b>JavaDataENG</b> folder.</li> <li>Open the incomplete class called <b>Question3.java</b> in the folders <b>Source Packages (src), Question3Package</b>.</li> <li>Add your examination number as a comment in the first line of the class (<b>Question3</b>).</li> </ul>

Do the following:

- Compile and execute the program. The program currently has no functionality. An example of the interface is given below.



Example of the output if the delivery was done by truck Tr4 and the actual fuel used is 85,0:

The screenshot shows a form titled "Fuel used". It contains a text input field with the value "85.0" and a label "Actual fuel used". Below this is a button labeled "Check fuel used". At the bottom of the form, there is a red error message box that reads: "ERROR: Difference in fuel used is too great".

(9)

## 2.2.5

### Button – [Calculate toll fees]

To determine the toll fees for the route used for the new delivery, the route number (for example RN3) must be entered in the text box provided. This route number must be sent as a parameter to the **determineTollFees** method.

The toll fees must be displayed in the label component provided. The amount must be displayed as currency (rand) to TWO decimal places.

Example of the output if the delivery was done by truck Tr4 on route RN3:

The screenshot shows a form titled "Toll fees". It contains a text input field with the value "RN3" and a label "Route to be followed". Below this is a button labeled "Calculate toll fees". At the bottom of the form, there is a label that reads: "Toll fees to be paid: R205.00".

(5)

- Ensure that your examination number is entered as a comment in the first line of the class as well as the form.
- Save all the files.
- Print code you created for both the classes, if printouts are required.

## 60 TOTAL SECTION B:



## 2.2.4

**Button – [Check fuel used]**

The driver of the truck is required to refill the fuel tank at the end of each delivery. Due to the nature of the routes, the amount of fuel used to refill the tank may sometimes differ from the fuel calculated for the delivery.

Write code to do the following:

- Enter the amount of fuel used to refill the tank in the text box provided.

**HINT:** The value of 75 can be used to test the code.

- Obtain the calculated amount of fuel used from the relevant attribute in the object class.

- Calculate the difference between the fuel used to refill the tank and the calculated fuel used as stored in the attribute of the **Delivery** object.

- If the difference is less than 10%, the **FuelUsed** attribute must be updated by setting its value to the new value as entered in the text box.

An appropriate message must be displayed in the text box provided, indicating whether the value of the **FuelUsed** attribute has been updated or not.

Example of the output if the delivery was done by truck Tr4 and the actual fuel used is 75,0:

The screenshot shows a Java Swing window titled "Fuel used". Inside the window, there is a text field containing the value "75.0". Below the text field is a button labeled "Check fuel used". At the bottom of the window, there is a label that reads "Fuel used changed from 70.4 to 75.0 litres".

- If the difference is greater than or equal to 10%, an error message should be displayed and the value of the **FuelUsed** attribute must not be updated.



Write code for the **New delivery** button to do the following:

- Use the new delivery number, the truck number, the odometer reading at the start and the odometer reading at the end of the delivery to instantiate the **Delivery** object.
- Display a message indicating that the object has been instantiated successfully.

Delivery object created successfully.

- Obtain the distance travelled by calling the **calculateDistance** method and calculate the estimated litres of fuel used for the delivery. Use the following information:

One litre of fuel is used for every five kilometres travelled.

Use the method you have written in QUESTION 2.1 to set the **FuelUsed/fuelUsed** attribute to this calculated value.

**NOTE:** The following buttons must be enabled:

- Check fuel used
- Calculate toll fees

## 2.2.3

### Button – [Display delivery]

Display the object in the output area provided using the **toString** method.

Example of the output for truck Tr4 with an end odometer reading of 2000:

Display delivery

Delivery number: 21  
Truck number: Tr4  
Odometer reading: (Start) 1648 (End) 2000  
Fuel used: 70.4 litres

(2)



- Display the **new delivery number** and **start odometer reading** in the text boxes provided.

- Display a suitable message in a dialog box and terminate the program if the text file cannot be accessed.

Example of the content of the relevant text boxes when truck Tr4 is selected:

**NOTE:** If you are unable to read the required information from the given text file, type in the number and the start odometer reading for the delivery into the text boxes to continue with the rest of the program.

(14)

## 2.2.2

### Button – [New delivery]

To instantiate a new **Delivery** object, the user needs to first enter the end odometer reading for the delivery in the textbox provided.

**NOTE:** The end odometer reading entered by the user must be greater than the previous odometer reading of the truck, because it is the start odometer reading for this delivery. No validation is necessary.

**NOTE:** The **Delivery** object variable has already been declared globally as part of the given code.



2.2

A text file named **DeliveryInfo.txt** contains an unknown number of lines of information on previously completed deliveries. Each line of information contains data on a single delivery in the following format:

**<delivery number>#<truck number>#<odometer reading on completion of the delivery>**

Example of some of the data in the text file named **DeliveryInfo.txt**:

```
1#Tr1#121110
2#Tr2#8010
3#Tr3#15021
4#Tr4#700
5#Tr1#121453
6#Tr3#15653
:
```

The data of the first delivery can be interpreted as follows:

- Delivery **1** identifies the delivery.
- Truck **Tr1** was used to make the delivery.
- The reading on the odometer on completion of the delivery was **121110**.

Do the following to complete the code for each button in the main form unit (Delphi)/GUI class (Java) as described below:

2.2.1 **Button – [Get data from file]**

- Select the specific truck to be used from the provided **Truck number** combo box.
- Use the given text file **DeliveryInfo.txt** to determine the following:

○ **New delivery number**

The delivery number for the new delivery will follow on the number of the last delivery stored in the text file. If the information of 20 deliveries are stored in the text file, the number of the new delivery will be 21.

○ **Start odometer reading**

The odometer reading for the last delivery that was made by the selected truck must be used as the start odometer reading for the new delivery.

Example:

If truck **Tr4** is selected and the odometer reading that was captured in the text file on completion of the last delivery for **Tr4** is 1648, then the value of 1648 must be used as the odometer reading at the start of the new delivery.



Complete the code in the given **delivery** class (**TDelivery/Delivery**) as described in QUESTION 2.1.1 to QUESTION 2.1.4 below:

2.1.1 Write code for a constructor method to receive the delivery number, truck number, odometer reading at the start of the delivery and odometer reading on completion of the delivery as parameter values. Assign these values to the relevant attributes of the object class. (3)

2.1.2 Write a mutator method and an accessor method for the **FuelUsed/fuelUsed** attribute. (4)

2.1.3 Write a method called **calculateDistance** to calculate and return the distance travelled based on the start and end odometer readings for the delivery. (3)

2.1.4 Different toll fees must be paid on different toll routes. The routes used are RN1, RN2, RN3 and RN4. Toll fees on these routes are dependent on the type of truck used. The company's trucks are classified as follows:

- Light-duty trucks: Tr1, Tr2
- Medium-duty truck: Tr3
- Heavy-duty trucks: Tr4, Tr5

A two-dimensional array called **tollFees** contains the toll fees for the different routes for different types of trucks and is supplied as part of the given code. The contents of the array can be represented as follows:

	Light-duty truck	Medium-duty truck	Heavy-duty truck
RN1	R105,50	R135,00	R210,00
RN2	R35,00	R54,00	R82,00
RN3	R85,00	R129,00	R205,00
RN4	R112,00	R170,00	R219,00

Rows: Represents the routes RN1 to RN4  
Columns: Represents the types of trucks

Write a method called **determineTollFees** to determine and return the toll fees to be paid for the delivery. The method must receive the route (RN1, RN2, RN3 or RN4) as a parameter. Use the two-dimensional array called **tollFees** to look up the toll fee for the route and type of truck that was used for the delivery.

**NOTE:** It is compulsory to use the given two-dimensional array in your solution to look up the toll fee. (10)





- Complete the code for this program as specified in QUESTION 2.1 and QUESTION 2.2 below.

2.1

The given incomplete object class (**TDelivery/Delivery**) contains the following code:

- The declaration of five attributes which describes a **delivery** object
- The declaration of a two-dimensional array to be used to determine toll fees
- A **toString** method

The attributes of a **delivery** object are the following:

Names of attributes		
<b>Delphi</b>	<b>Java</b>	<b>Description</b>
fDeliveryNum	deliveryNum	Number assigned to a specific delivery, for example 1, 2, et cetera
fTruckNum	truckNum	Truck number, for example Tr1, Tr2, Tr3, Tr4 or Tr5
fFuelUsed	fuelUsed	Fuel used during the delivery
fOdoStart	odoStart	Odometer reading at the start of the delivery
fOdoEnd	odoEnd	Odometer reading on completion of the delivery



SECTION B

QUESTION 2: OBJECT-ORIENTATED PROGRAMMING

SuperTrans Courier Services owns five trucks. The trucks are classified as light, medium or heavy-duty trucks. The trucks are used on four different routes (RN1 to RN4) to deliver cargo.

INSTRUCTIONS:

<b>Delphi programmers</b>	<p>The project <b>Question2</b> is provided in the <b>DelphiDataENG</b> folder which contains:</p> <ul style="list-style-type: none"><li>○ A main form unit called <b>Question2_U.pas</b></li><li>○ An incomplete unit file called <b>Delivery_U.pas</b></li><li>○ A text file (<b>DeliveryInfo.txt</b>) that contains information on deliveries</li></ul> <p>Open the incomplete project file called <b>Question2_P.dpr</b> in the <b>Question2</b> folder.</p> <ul style="list-style-type: none"><li>• View (Ctrl + F12) the unit file <b>Delivery_U.pas</b> and add your examination number as a comment in the first line of both files</li></ul> <p><b>Question2_U.pas</b> and <b>Delivery_U.pas</b>.</p>
<b>Java programmers</b>	<p>The project <b>Question2</b> is provided to you in the <b>JavaDataENG</b> folder which contains:</p> <ul style="list-style-type: none"><li>○ A GUI class file called <b>Question2.java</b></li><li>○ An incomplete object class file called <b>Delivery.java</b></li><li>○ A text file (<b>DeliveryInfo.txt</b>) that contains information on deliveries</li></ul> <p>Open the incomplete classes called <b>Question2.java</b> and <b>Delivery.java</b> in the folders <b>Source Packages (src)</b>, <b>Question2Package</b>.</p> <ul style="list-style-type: none"><li>• Add your examination number as a comment in the first line of both classes <b>Question2.java</b> and <b>Delivery.java</b>.</li></ul>

Do the following:

- Compile and execute the program. The program currently has no functionality. An example of the interface is shown on the next page:



Example of the contents of the output area if the city of Durban is selected:

**Question 1.5**

Select a city:

```

Durban
2014-12-01 Durban to Cape Town
2014-12-04 Port Elizabeth to Durban
2014-12-05 Durban to Kimberley
2014-12-05 Durban to Potchefstroom
2014-12-07 Potchefstroom to Durban
2014-12-09 Durban to Bloemfontein
  
```

Example of the contents of the **December2014Durban.txt** text file:

```

2014-12-01 Durban to Cape Town
2014-12-04 Port Elizabeth to Durban
2014-12-05 Durban to Kimberley
2014-12-05 Durban to Potchefstroom
2014-12-07 Potchefstroom to Durban
2014-12-09 Durban to Bloemfontein
  
```

(12)

- Enter your examination number as a comment in the first line of the program file.
- Save your program.
- A printout of the code may be required.

**TOTAL SECTION A: 50**



1.5

**Button – [View and save deliveries]**

All deliveries for December 2014 are stored in the given array called **arrDecDeliveries**. The format of each entry in the array is as follows:

<date><space><place of departure><space><space><destination>

Example:

2014-12-01 Durban to Cape Town

The user has to select a city from the provided combo box. All the deliveries that were made during December 2014 to or from the selected city must be displayed in the provided output area and also written to a text file.

Write code to do the following:

- Create a text file where the name of the file is made up of the text 'December2014' combined with the name of the city that was selected.

Example:

If Durban was selected, the name of the text file must be **December2014Durban.txt**.

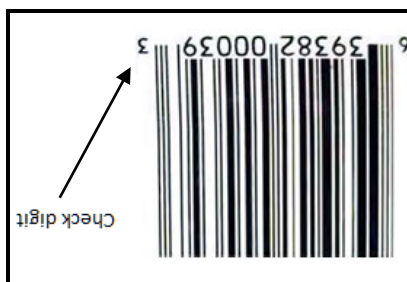
- Use the data stored in the **arrDecDeliveries** array and display the deliveries to and from the selected city in the output area provided. Use the name of the city as a heading.

- Store the deliveries to and from the selected city in the text file that was created – one delivery per line.

Example of a possible output for an invalid bar code:

(14)





The bar code number consists of twelve digits. For example, the bar code number shown in the picture above is 639382000393. The last digit of a UPC is called a check digit. The check digit is used by a scanner to determine whether a bar code is valid or not.

Use the algorithm below to write code for the **Validate bar code** button. The code needs to verify the check digit and display a message to indicate whether the bar code is valid or not.

#### Algorithm:

```

1. barCode ← text box value
2. sumOddPositions ← 0
3. sumEvenPositions ← 0
4. loop from first digit to second last digit of barCode
   if the logical position of the digit in the bar code is even
     sumEvenPositions ← sumEvenPositions + digit at position
   else
     sumOddPositions ← sumOddPositions + digit at position
6. sum ← sumOddPositions * 3 + sumEvenPositions
7. checkDigit ← 10 – (sum modulus 10)
8. if checkDigit = last digit of barCode
   display the check digit and a suitable message indicating that
   the bar code is valid
else
   display a suitable message indicating that the bar code is invalid

```

Example:

If the bar code is 639382000393 then:

$$\begin{aligned} \text{sumOddPositions} &= 6+9+8+0+0+9 = 32 \\ \text{sumEvenPositions} &= 3+3+2+0+3 = 11 \end{aligned}$$

Example of a possible output for a valid bar code:



**Question 1.2**

Select the category

A1 (0kg - 1kg)
A2 (>1kg - 2kg)
A3 (>2kg - 5kg)
A4 (>5kg)

☒ Speed post

Tick if speed post is required

**Question 1.3**

Delivery cost

R735.00

Delivery box number

4

Example of output when speed delivery is NOT required in QUESTION 1.2:

**Question 1.2**

Select the category

A1 (0kg - 1kg)
A2 (>1kg - 2kg)
A3 (>2kg - 5kg)
A4 (>5kg)

☐ Speed post

Tick if speed post is required

**Question 1.3**

Delivery cost

R635.00

Delivery box number

3

**NOTE:** Due to the nature of the random function the value for the delivery box displayed in the screenshot above may differ from the value displayed by your program.

(9)

1.4

**Button – [Validate bar code]**

A Universal Product Code (UPC) bar code is printed on items to be delivered. The picture on the next page shows an example of a UPC bar code:



Please turn over

Copyright reserved

- A check box that indicates whether speed post must be used. A standard amount of R100,00 is charged for speed post.

When the user clicks on the **Delivery cost** button the number of kilometres entered in QUESTION 1.1, the selected weight category and whether speed post is required or not must be used to calculate the delivery cost.

Example of the output if the number of kilometres is 635, the weight of the item to be delivered is in the A2 category and speed post is required:

### Button – [Delivery box number]

1.3

Items to be delivered need to be placed in specific delivery boxes. The correct delivery box for each individual item must be determined using the criteria below:

- There are five delivery boxes numbered from 1 to 5.
- All **speed post items** will be placed in delivery box 4.
- All the other delivery items will be **randomly** placed in the remaining delivery boxes (1, 2, 3 or 5).

Display the number of the delivery box into which the item must be placed.

Example of output if speed post was requested in QUESTION 1.2 (on the next page):



- Complete the code for each section of QUESTION 1 as described in QUESTIONS 1.1 to QUESTION 1.5 below.

1.1 Button – [Confirm Delivery]

Obtain the following data from the relevant components:

- Place of departure from the **Place of Departure** combo box
- Destination from the **Destination** combo box
- Number of kilometres from the **kilometres** text box

Create a line of text as output that indicates the place of departure, the destination and the number of kilometres as shown in the example below. Place the constructed line of text in the label component provided.

Example of possible input:

Select the place of departure	Johannesburg
Select the destination	Durban
Enter the number of kilometres	635

Required output:

Delivery label
Johannesburg to Durban : 635 km

1.2 Button – [Delivery cost]

The following components are provided:

- A list box that indicates the weight categories of deliveries in terms of codes (A1–A4) in the following format:

<code><space><(range in kg)>

A1 (0kg - 1kg)
A2 (>1kg - 2kg)
A3 (>2kg - 5kg)
A4 (>5kg)

The following tariffs per weight category apply:

Category	Tariff per kilometre
A1	R0,60
A2	R1,00
A3	R1,25
A4	R1,65





**SCENARIO:**

SuperTrans Courier Services is a national transport company with branches throughout South Africa. You are requested to assist with some of the software applications the company intends to implement shortly.

**SECTION A****QUESTION 1: GENERAL PROGRAMMING SKILLS****INSTRUCTIONS:**

Delphi programmers	Java programmers
<ul style="list-style-type: none"> <li>The project <b>Question1</b> is provided in the <b>DelphiDataENG</b> folder.</li> <li>Open the incomplete project file <b>Question1_P.dpr</b> in the <b>Question1</b> folder.</li> <li>Add your examination number as a comment in the first line of the main form unit (<b>Question1_U</b>) file.</li> </ul>	<ul style="list-style-type: none"> <li>The project <b>Question1</b> is provided in the <b>JavaDataENG</b> folder.</li> <li>Open the incomplete class called <b>Question1.java</b> contained in the folders <b>Source Packages (src)</b>, <b>Question1Package</b>.</li> <li>Add your examination number as a comment in the first line of the class (<b>Question1</b>).</li> </ul>

**Do the following:**

- Compile and execute the program. The interface displays five different sections labelled Question 1.1 to Question 1.5. The program currently has no functionality. An example of the interface is given below:



13. The files you need to complete this question paper have been given to you on a disk/CD/DVD/flash disk or the disk space allocated to you in the form of a password-protected executable file:

- Delphi learners must use the file **DelphiDataENG.exe**
- Java learners must use the file **JavaDataENG.exe**

Do the following:

- Double click on the file
- Click on the extract button
- Enter the following password: **Transport@(\$**

**List of files provided in the folder DelphiDataENG/JavaDataENG (once extracted):**

#### **Delphi files**

**Question1:**  
Question1\_P.dpr  
Question1\_P.res  
Question1\_U.dfm  
Question1\_U.pas

#### **Question2:**

Delivery\_U.pas  
DeliveryInfo.txt  
Question2\_P.dpr  
Question2\_P.res  
Question2\_U.dfm  
Question2\_U.pas

#### **Question3:**

Question3\_P.dpr  
Question3\_P.res  
Question3\_U.dfm  
Question3\_U.pas

#### **Java (Netbeans) files**

**Question1:**  
Question1.form  
Question1.java

#### **Question2:**

Delivery.java  
DeliveryInfo.txt  
Question2.form  
Question2.java

#### **Question3:**

Question3.form  
Question3.java



**INSTRUCTIONS AND INFORMATION**

1. This paper is divided into THREE sections. Candidates must answer ALL THREE sections.
2. The duration of this examination is three hours. Because of the nature of this examination it is important to note that you will not be permitted to leave the examination room before the end of the examination session.
3. This paper is set in programming terms that are not specific to any particular programming language (Delphi/Java (using the Netbeans IDE)).
4. Make sure that you answer the questions according to the specifications that are given in each question. Marks will be awarded according to the set requirements only.
5. Answer only what is asked in each question. For example, if the question does not ask for data validation, then no marks will be awarded for data validation.
6. Your programs must be coded in such a way that they will work with any data and not just the sample data supplied or any data extracts that appear in the question paper.
7. Routines such as search, sort and selection must be developed from first principles. You may not use the built-in features of a programming language for any of these routines.
8. All data structures must be defined by you, the programmer. You may not use components provided within the interface to store and later retrieve data.
9. You must save your work regularly on the disk you have been given, or the disk space allocated to you for this examination session.
10. Make sure that your examination number appears as a comment in every program that you code as well as on every event indicated.
11. If required, print the programming code of all the programs/classes that you completed. You will be given half an hour printing time after the examination session.
12. At the end of this examination session you must hand in a disk/CD/DVD/flash disc with all your work saved on it OR you must make sure that all your work has been saved on the disk space allocated to you for this examination session. Ensure that all files can be read.





# MORNING SESSION

This question paper consists of 22 pages.

TIME: 3 hours

MARKS: 150

NOVEMBER 2014

INFORMATION TECHNOLOGY P1

INFT.1

GRADE 12

NATIONAL  
SENIOR CERTIFICATE

Department:  
Basic Education  
REPUBLIC OF SOUTH AFRICA

basic education

