NASIONALE SENIOR SERTIFIKAAT-EKSAMEN
NOVEMBER 2023

INLIGTINGSTEGNOLOGIE: VRAESTEL I

NASIENRIGLYNE

Tyd: 3 uur | 150 punte

Hierdie nasienriglyne is opgestel vir gebruik deur eksaminators en hulpeksaminators van wie verwag word om almal 'n standaardiseringsvergadering by te woon om te verseker dat die riglyne konsekwent vertolk en toegepas word by die nasien van kandidate se skrifte.

Die IEB sal geen bespreking of korrespondensie oor enige nasienriglyne voer nie. Ons erken dat daar verskillende standpunte oor sommige aangeleenthede van beklemtoning of detail in die riglyne kan wees. Ons erken ook dat daar sonder die voordeel van die bywoning van 'n standaardiseringsvergadering verskillende vertolkings van die toepassing van die nasienriglyne kan wees.

AFDELING A SQL

VRAAG 1.1 [4]

```
SELECT *
FROM tblParkerings
WHERE PremieParkering = true
ORDER BY DaaglikseTarief
```

VRAAG 1.2 [5]

```
SELECT*
FROM tblMotors
WHERE MotorRegistrasie LIKE '*GP'
AND MID(MotorRegistrasie, 3,1) = ' '
```

Aanvaar % vir mysql

```
JAVADB
AND SUBSTR (MotorRegistrasie, 3,1) = ' '
```

VRAAG 1.3 [6]

```
SELECT Adres, DaaglikseTarief, DatumBygevoeg, PremieParkering
FROM tblParkerings
WHERE PremieParkering = True OR
(DaaglikseTarief>135 AND YEAR (DatumBygevoeg) = 2022)
```

VRAAG 1.4 [5]

```
SELECT *
FROM tblParkerings
WHERE DaaglikseTarief = (SELECT MAX(DaaglikseTarief)
                    FROM tblParkerings)
```

VRAAG 1.5 [4]

```
SELECT  tblMotors.MotorRegistrasie, Model
FROM tblMotors
LEFT JOIN tblVerhuurdeParkerings
ON tblVerhuurdeParkerings.MotorRegistrasie =
tblMotors.MotorRegistrasie
WHERE tblVerhuurdeParkerings.MotorRegistrasie IS NULL
```

Aanvaar ParkeringID, BeginDatum OR EindDatum IS NULL

```
SELECT *
FROM tblMotors
WHERE tblMotors.MotorRegistrasie
          NOT IN (SELECT tblVerhuurdeParkerings.MotorRegistrasie
                   FROM tblVerhuurdeParkerings)
```

VRAAG 1.6 [7]

```
SELECT Adres , Count(*) AS KereVerhuur
FROM tblParkerings, tblVerhuurdeParkerings
WHERE tblParkerings.ParkeringID = tblVerhuurdeParkerings.ParkeringID
GROUP BY Adres
HAVING Count(*) > 2
```

Aanvaar

```
HAVING Count(*) >= 3
```

MySQL

```
HAVING KereVerhuur > 2
HAVING KereVerhuur >= 3
```

VRAAG 1.7 [9]

```
SELECT tblMotors.MotorRegistrasie, Eienaar, DaaglikseTarief,
BeginDatum, EindDatum,  {fn TIMESTAMPDIFF
(SQL_TSI_DAY, BeginDatum, EindDatum)} * DaaglikseTarief AS
HuurBedrag
FROM tblMotors, tblVerhuurdeParkerings, tblParkerings
WHERE
tblMotors.MotorRegistrasie=tblVerhuurdeParkerings.MotorRegistrasie
AND tblParkerings.ParkeringID=tblVerhuurdeParkerings.ParkeringID;
```

JavaDB

```
SELECT tblMotors.MotorRegistrasie, Eienaar, DaaglikseTarief,
BeginDatum, EindDatum,  (EindDatum - BeginDatum)
* DaaglikseTarief AS HuurBedrag
FROM tblMotors, tblVerhuurdeParkerings, tblParkerings
WHERE
tblMotors.MotorRegistrasie=tblVerhuurdeParkerings.MotorRegistrasie
AND tblParkerings.ParkeringID=tblVerhuurdeParkerings.ParkeringID;
```

VRAAG 1.8 [6]

```
UPDATE tblMotors
SET Model = 'VW' & RIGHT(Model, LEN(Model)-11)
WHERE Model LIKE 'Volkswagen*'
```

Aanvaar

```
SET Model = 'VW' & RIGHT(Model, LEN(Model)-10)
WHERE Model LIKE 'Volkswagen%'
```

JavaDB

```
SET Model = 'VW' || SUBSTR(Model, 11, LENGTH(MODEL))
```

VRAAG 1.9 [4]

```
DELETE *
FROM tblParkerings
WHERE DatumBygevoeg < #2018/01/01#
```

Aanvaar

```
WHERE YEAR(DatumBygevoeg) < 2018
```

AFDELING B OBJEKGEORIËNTEERDE PROGRAMMERING

JAVA-OPLOSSING

VRAAG 2 Daagliks.java

```
//V2.1 - 4
// klasopskrif
public class Daagliks
{
    // alle velde privaat
    // korrek getipeer
    // korrek benoem
    private String registrasieNommer;
    private LocalDateTime aankomsDatumTyd;
    private LocalDateTime vertrekDatumTyd;

    //V2.2 - 2
    // velde verklaar as openbaar
    // en finaal
    //korrek benoem en toegeken
    public static final double UURTARIEF = 24.50;
    public static final double BOETEBEDRAG = 850;

    //V2.3 - 3
    // opskrif korrek
    // parameters korrek benoem en getipeer
    public Daagliks(String inRN, LocalDateTime inEY,
                LocalDateTime inET)
    {
        // velde korrek toegeken
        registrasieNommer = inRN;
        aankomsDatumTyd = inEY;
        vertrekDatumTyd = inET;
    }

    //V2.4 - 2
    // Alle krymetodes korrek benoem
    // Alle terugsendings korrekte tipe
    public String kryRegistrasieNommer()
    {
        return registrasieNommer;
    }

    public LocalDateTime kryAankomsDatumTyd()
    {
        return aankomsDatumTyd;
    }

    public LocalDateTime kryVertrekDatumTyd()
    {
        return vertrekDatumTyd;
    }
```

```
    //V2.5 - 8
    public double kryParkeerGeld()
    {

        // kontroleer of dae verskillend is
        // skakel DateTime om in datum
                //- aanvaar enige korrekte alternatiewe antwoord
                // moontlike alternatief
                // if (aankomsDatumTyd.getDayOfYear() !=
                // vertrekDatumTyd.getDayOfYear()) {
        if (!aankomsDatumTyd.toLocalDate().equals
           (vertrekDatumTyd.toLocalDate()))
        {
            // return BOETEBEDRAG
            return BOETEBEDRAG;
        } else
        {
            // kry tydverskil
            // skakel beide DatumTyd-objekte om in Tyd-objekte
            // skep toepaslike objek deur verskil te gebruik
            Duration verskil = Duration.between
                           (aankomsDatumTyd.toLocalTime(),
                            vertrekDatumTyd.toLocalTime());
            long secs = verskil.toSeconds();
            LocalTime rVerskil = LocalTime.ofSecondOfDay(secs);

            // bereken fooi deur UURTARIEF te gebruik
            double fooi = UURTARIEF * rVerskil.getHour();
            // stuur fooi terug
            return fooi;
        }
    }

    //V2.6 - 4
    // opskrif en terugsending korrek
    public String naString()
    {
        // bevat alle velde
        // formatering korrek
        // roep kryParkeerGeld
        return "Registrasie: " + registrasieNommer +
               "\tFooi: R" + kryParkeerGeld() +
               "\nAankoms: " + aankomsDatumTyd +
               " Vertrek: " + vertrekDatumTyd ;
    }

}
```

VRAAG 3 LangTermyn.java

```
//V3.1 - 2
// klas korrek benoem
// brei Daagliks uit
public class LangTermyn extends Daagliks {
    //V3.2 - 1
    // veld korrek getipeer en benoem
    private String parkeerPlek;

    //V3.3 - 1
    // veld as openbaar, finaal verklaar, korrek getipeer en benoem
    public static final double TARIEFPERDAG = 250;

    //V3.4 - 4
    // konstruktoropskrif korrek
    public LangTermyn(String inRN, LocalDateTime inEY,
LocalDateTime inET , String inPB)
    {
        // ouerkonstruktor geroep
        // korrekte parameters gegee
        super(inRN, inEY, inET);

        // dogterklasveld toegeken deur parameters te gebruik
        parkeerPlek = inPB;
    }

    //V3.5 - 1
    // opskrifte korrek en stuur korrekte veld terug
    public String kryParkeerPlek()
    {
        return parkeerPlek;
    }

    //V3.6 - 5
    // korrekte opskrifnaam en stuur dubbel terug
    public double kryParkeerGeld()
    {
        // skakel velde om in Datum-objekte
        // bepaal die verskil in datums
        Period verskil = Period.between(
                     kryAankomsDatumTyd().toLocalDate(),
                     kryVertrekDatumTyd().toLocalDate());
        // berekening korrek deur TARIEFPERDAG te gebruik
        // gebruik korrekte metode vir toegang tot die getal dae
        double fooi = verskil.getDays() * TARIEFPERDAG;
        return fooi;
    }
```

```
    //V3.7 - 3
    // opskrif korrek en stuur string terug
    public String naString()
    {
        // roep naString uit ouer
        // las veld aan
        return super.naString() + "\nParkeerplek: " + parkeerPlek;
    }
}
```

VRAAG 4 EN 6.1, 6.2 ParkeringBestuurder.java

```
//V4.1 - 1
// klasopskrif korrek
public class ParkeringBestuurder {

    //V4.2 - 4
    // velde privaat
    // skikking van 50 Daagliks
    // tipeer ouerklas
    // grootte-veld korrekte tipe
    private Daagliks pSkik[] = new Daagliks[50];
    private int grootte = 0;

    //V4.3 - 11
    // konstruktor
    public ParkeringBestuurder() {
        try {
            // maak lêer oop en lus daardeur
            String reg, entSt, exSt, bay = "";
            Scanner scFile = new Scanner(new
                                File("parkerings.txt"));
            while (scFile.hasNextLine()) {
                // lees reël uit tekslêer
                String line = scFile.nextLine();
                // onttrek eerste 3 velde uit tekslêer
                Scanner scLine = new Scanner(line).
                                 useDelimiter(";");
                reg = scLine.next();
                entSt = scLine.next();
                exSt = scLine.next();
                // gebruik korrekte DatumTyd-formaat uit tekslêer
                DateTimeFormatter format =
                                  DateTimeFormatter.ofPattern
                                  ("yyyy/MM/dd HH:mm");
                // skep DatumTyd-objekte
                LocalDateTime entry = LocalDateTime.parse
                                      (entSt, format);
                LocalDateTime exit = LocalDateTime.parse
                                     (exSt, format);
```

```
                // kontroleer vir langtermynparkeerplek
                if (scLine.hasNext()) {
                    // kry parkeerplek
                    bay = scLine.next();
                    // skep LangTermyn-objek en
                    // voeg by skikking by
                    pSkik[grootte] = new LangTermyn
                                        (reg, entry, exit, bay);
                } else {
                    // skep Daagliks-objek en voeg by skikking by
                    pSkik[grootte] = new Daagliks(reg, entry, exit);
                }
                // inkrementeer grootte
                grootte++;
                scLine.close();
            }
            scFile.close();
        } catch (FileNotFoundException e) {
            System.out.println("Lêer ontbreek");
        }
    }

    //V4.4 - 4
    // opskrif korrek en stuur string terug
    public String naString() {
        String r = "";
        // lus deur skikking
        for (int i = 0; i < grootte; i++) {
            // las by string aan
            // voeg oop reël by
            r += pSkik[i].naString() + "\n\n";
        }
        return r;
    }

    //V4.5 - 6
    public void sorteerVolgensRegistrasie() {
        // buitenste for-lus korrek
        for (int i = 0; i < grootte - 1; i++) {
            // binneste for-lus korrek
            for (int j = i + 1; j < grootte; j++) {
                // vergelyk die korrekte elemente
                // deur registrasienommer te gebruik
                if (pSkik[i].kryRegistrasieNommer().
                    compareToIgnoreCase
                    (pSkik[j].kryRegistrasieNommer()) > 0) {
                    // stoor in temp met korrekte tipe
                    Daagliks temp = pSkik[j];
                    // ruil elemente om
                    pSkik[j] = pSkik[i];
                    pSkik[i] = temp;
                }
            }
        }
```

```
    }
    //V6.1 - 6
    // korrekte metodeopskrif en terugsending
    public String kryRegistrasieLys() {
        String temp = "";
        //lus deur skikking
        for (int i = 0; i < grootte; i++) {
            // kontroleer of motorregistrasie
            // reeds in temp is
            if (!temp.contains(pSkik[i].kryRegistrasieNommer())) {
                // voeg registrasie by lys by
                // voeg 'n #-karakter by
                temp += pSkik[i].kryRegistrasieNommer() + "#";
            }
        }
        return temp;
    }

//V6.2 - 1
// metodeopskrif en terugsending korrek
    public String genereerRekening() {
        //V6.2.1 - 1
        // roep kryRegistrasieLys en ken aan motors toe
        String motors = kryRegistrasieLys();

        String bay = "", output = "";
        LangTermyn temp = null;

        //V6.2.2
        //isoleer registrasienommers - 3
        // maak skandeerder oop
        // Alternatiewe metode – gebruik verdeling om in 'n
skikking om te skakel
        Scanner scMotors = new Scanner(motors).useDelimiter("#");
        String currentReg;
        // kontroleer vir nog registrasienommers
        // lus deur skikking tot einde by array.length-veld
        while (scMotors.hasNext()) {
            // onttrek registrasienommer
            // verkry toegang tot skikkingselement
            currentReg = scMotors.next();

            //skep 'n string - 3*
            // *las die registrasienommer voor die inskrywings aan
            output = output + currentReg + "\n";

            //soek en voeg by totaal by - 5
            // inisialiseer totaal binne die lus
            double totaal = 0;
            boolean Daagliks = false;
            // lus deur skikking
            for (int i = 0; i < grootte; i++) {
```

```
                // kontroleer of motorregistrasie reeds in lys is
                if (currentReg.equals(pSkik[i].
                    kryRegistrasieNommer())) {
                    // voeg parkeergeld by
                    // totaal by
                    totaal += pSkik[i].kryParkeerGeld();

                    // *las besonderhede van elke inskrywing by
                        afvoer by
                    output += pSkik[i].kryAankomsDatumTyd() + " "
                            + pSkik[i].kryVertrekDatumTyd() + " "
                            + pSkik[i].kryParkeerGeld() + "\n";

                    //kontroleer of langtermynparkeerplek - 4
                    // kontroleer of pSkik[i]
                    // 'n LangTermyn-objek is
                    if (pSkik[i] instanceof LangTermyn) {
                        Daagliks = false;
                        // ken pSkik[i] aan LangTermyn-objek toe
                        temp = (LangTermyn) pSkik[i];
                        // teken parkeerpleknommer aan
                        bay = temp.kryParkeerPlek();
                    } else {
                        Daagliks = true;
                        bay = "";
                    }
                }
            }
            // *las die parkeergeld aan
            output += "Totale parkeergeld verskuldig: R" + totaal + "\n";

            //voeg die parkeerpleknommer by - 2
            // kontroleer of die registrasie LangTermyn-objek is
            if (Daagliks == false) {
                // las die parkeerpleknommer aan
                output += "Nommer van langtermynparkeerplek: " + bay + "\n\n";

                //voeg boodskap by - 2
                // kontroleer of die totaal bo 1500 is
            } else if (totaal > 1500 && Daagliks == true) {
                // las die boodskap aan
                output += "Oorweeg dit om langtermynparkering te huur
                            parkering\n\n";
            }
            else output += "\n\n";
        }
        return output;
    }
}
```

VRAAG 5, 6.3 ParkeringGK.java

```
//V5.1 - 1
// klasopskrif korrek
public class ParkeringGK
{
    public static void main(String[] args)
    {
        //V5.2 - 2
        // ParkeringBestuurder-objek geskep
        // op korrekte plek
        ParkeringBestuurder pb = new ParkeringBestuurder();

        //V5.3 - 2
        // sorteermetode geroep
        pb.sorteerVolgensRegistrasie();
        // naString-metode geroep
        System.out.println(pb.naString());

        //V6.3 - 2
        // korrekte metodenaam
        // in 'n afvoerstelling (getipeerde metoderoep)
        System.out.println(pb.genereerRekening());
    }
}
```

DELPHI-OPLOSSING

VRAAG 2 uDaagliks.pas

```
unit uDaagliks;

interface
uses SysUtils, DateUtils;
 //V2.1 - 4
//klasopskrif
type TDaagliks = class
  //alle velde privaat
  //korrek getipeer
  //korrek benoem
  private
    registrasieNommer : string;
    aankomsDatumTyd : TDateTime;
    vertrekDatumTyd : TDateTime;


  public
     //V2.2 - 2
    //velde verklaar as openbaar en finaal
    //korrek benoem en toegeken
    const
      UURTARIEF = 24.50;
      BOETEBEDRAG = 850;

    constructor Create( inRN:string ; inEY: TDateTime; inET:
TDateTime ) ;
    function kryRegistrasieNommer() : string;
    function kryAankomsDatumTyd() : TDateTime;
    function kryVertrekDatumTyd() : TDateTIME;
    function kryParkeerGeld() : double; virtual;
    function naString() : string; virtual;

end;

implementation

{ TDaagliks }
//V2.3 - 3
//opskrif korrek
//parameters korrek benoem en getipeer
constructor TDaagliks.Create(inRN: string; inEY, inET: TDateTime);
begin
     //velde korrek toegeken
    registrasieNommer := inRN;
    aankomsDatumTyd := inEY;
    vertrekDatumTyd := inET
end;
```

```
//V2.4 - 2
//Alle krymetodes korrek benoem
//Alle terugsendings korrekte tipe
function TDaagliks.kryAankomsDatumTyd: TDateTime;
begin
  Result := aankomsDatumTyd;
end;

function TDaagliks.kryVertrekDatumTyd: TDateTIME;
begin
  Result := vertrekDatumTyd;
end;

function TDaagliks.kryRegistrasieNommer: string;
begin
  Result:= registrasieNommer;
end;

//V2.5 - 8
function TDaagliks.kryParkeerGeld: double;
var
  fooi : double;
  sBetween, hBetween , mBetween : integer;
begin
   //kontroleer of dae verskillend is
   //skakel DatumTyd om in datum
   //aanvaar enige korrekte alternatiewe antwoord
    if NOT(DayOfTheYear(aankomsDatumTyd) =
DayOfTheYear(vertrekDatumTyd)) then
    begin
     //return BOETEBEDRAG
         Result := BOETEBEDRAG;
    end
    else
    begin
         //kry tydverskil
         //skakel beide DatumTyd-objekte om in Tyd-objekte
         //skakel om in ure
         fooi:= 0;

         sBetween := SecondsBetween( aankomsDatumTyd ,
vertrekDatumTyd);
         hBetween := sBetween div 3600;
         sBetween := sBetween - hBetween * 3600;
         mBetween := sBetween div 60;

          //bereken fooi deur UURTARIEF te gebruik
         fooi := UURTARIEF * hBetween;
           // stuur fooi terug
         Result := fooi;
    end;

end;
```

```
//V2.6 - 4
//opskrif en terugsending korrek
function TDaagliks.naString: string;
begin
   //bevat alle velde
  //formatering korrek
  //roep kryParkeerGeld
   Result := 'Registrasie: ' + registrasieNommer + #13#10 +
'Aankoms: ' + DateTimeToStr( aankomsDatumTyd ) + #13#10 +
'Vertrek: ' + DateTimeToStr( vertrekDatumTyd);
end;

end.
```

VRAAG 3 uLangTermyn.pas

```
unit uLangTermyn;

interface
 uses SysUtils, DateUtils, uDaagliks;
 //V3.1 - 2
//klas korrek benoem
//brei Daagliks uit
 type TLangTermyn = class(TDaagliks)
  //V3.2 - 1
    //veld korrek getipeer en benoem
   private
    parkeerPlek : string;

   public
    //V3.3 - 1
    //veld verklaar as openbaar, finaal, korrek getipeer en benoem
    const
      TARIEFPERDAG = 250.0;

    constructor Create(inRN:string ; inEY: TDateTime; inET:
TDateTime; inPB : string);
    function kryParkeerPlek() : string;
    function kryParkeerGeld() : double ; override;
    function naString() : string ; override;
 end;

implementation

{ TLangTermyn }
```

```
//V3.4 - 4
//konstruktoropskrif korrek
constructor TLangTermyn.Create(inRN: string; inEY : TDateTime;
inET: TDateTime; inPB: string);
begin
  //ouerkonstruktor geroep
  //korrekte parameters gegee
  Inherited Create( inRN, inEY, inET);
    //dogterklasveld toegeken deur parameters te gebruik
  parkeerPlek := inPB;
end;
//V3.5 - 1
//opskrifte korrek en stuur korrekte veld terug
function TLangTermyn.kryParkeerPlek: string;
begin
  Result := parkeerPlek;
end;

  //V3.6 - 5
    //korrekte opskrifnaam en stuur reëel/dubbel terug
function TLangTermyn.kryParkeerGeld: double;
var
  verskil : integer;
  fooi : real;
begin
   //skakel velde om in Datum-objekte
   //bepaal die verskil in datums
   verskil:= DaysBetween(kryAankomsDatumTyd(),
kryVertrekDatumTyd);
    //berekening korrek deur TARIEFPERDAG te gebruik
    //gebruik korrekte metode vir toegang tot die getal dae
   fooi := verskil * TARIEFPERDAG;

   Result := fooi;
end;

//V3.7 - 3
//opskrif korrek en stuur string terug
function TLangTermyn.naString: string;
begin
   //roep naString van ouer
   //las veld aan
  Result := Inherited naString + '' + #13#10 + 'ParkeerPlek ' +
parkeerPlek;
end;

end.
```

VRAAG 4 EN 6.1, 6.2 uParkeringBestuurder.pas

```
unit uParkeringBestuurder;

interface
uses SysUtils, DateUtils, uDaagliks, uLangTermyn;
//V4.1 - 1
//klasopskrif korrek
type tParkeringBestuurder = class
    //V4.2 - 4
  //velde privaat
  //skikking van 50 Daagliks
  //tipeer ouerklas
  //grootte-veld korrekte tipe
  private
    pSkik : array[1..50] of tDaagliks;
    grootte : integer;

  public
    constructor Create();
    function naString : string;
    procedure sorteerVolgensRegistrasie();
    function kryRegistrasieLys() : string;
    function genereerRekening(): string;
end;

implementation

{ tParkeringBestuurder }

//V4.3 - 11
//konstruktor

constructor tParkeringBestuurder.Create;
var
  inFile : textfile;
  line, reg, entSt, exSt, bay , date , d , m , y , h , mi,d2 , m2
, y2 , h2 , mi2  : string;
  entryDT, exitDT : TDateTime;
begin
//maak lêer oop en lus deur lêer
  if FileExists('parkerings.txt') <> true then
    begin
      WriteLn('Lêer ontbreek');
    end
  else
    begin
      AssignFile(inFile, 'parkerings.txt');
      Reset(inFile);

      grootte:=0;

      while NOT EOF(inFile) do
        begin
```

```
        //lees reël uit tekslêer
          ReadLn(inFile, line);
       //inkrementeer grootte
          Inc(grootte);

          //onttrek eerste 3 velde
          //gebruik korrekte DatumTyd-formaat
          reg := Copy(line , 1 , Pos(';', line) - 1);
          Delete(line , 1 , Pos(';',line));

          entST := Copy(line , 1 , Pos(';', line) - 1);
          Delete(line , 1 , Pos(';',line));

          exST := line;


          y := Copy(entSt,1 , Pos('/', entSt) - 1);
          Delete(entSt,1 , Pos('/', entSt));

          m := Copy(entSt,1 , Pos('/', entSt) - 1);
          Delete(entSt,1 , Pos('/', entSt));

          d := Copy(entSt,1 , Pos(' ', entSt) - 1);
          Delete(entSt,1 , Pos(' ', entSt));

          h := Copy(entSt,1 , Pos(':', entSt) - 1);
          Delete(entSt,1 , Pos(':', entSt));
          mi := entSt;
           //skep DatumTyd-objekte
          entryDT := EncodeDateTime(StrToInt(y), StrToInt(m) ,
StrToInt(d) , StrToInt(h) , StrToInt(m), 0 , 0);


          y := Copy(exST ,1 , Pos('/', exST ) - 1);
          Delete(exST ,1 , Pos('/', exST ));

          m := Copy(exST ,1 , Pos('/', exST ) - 1);
          Delete(exST ,1 , Pos('/', exST ));

          d := Copy(exST ,1 , Pos(' ', exST ) - 1);
          Delete(exST ,1 , Pos(' ', exST ));

          h := Copy(exST ,1 , Pos(':', exST ) - 1);
          Delete(exST ,1 , Pos(':', exST ));
          mi := exST;


          exitDT := EncodeDateTime(StrToInt(y), StrToInt(m) ,
StrToInt(d) , StrToInt(h) , StrToInt(m), 0 , 0);
            //kontroleer vir langtermynparkeerplek
          if Pos(';', line) > 0 then
          begin
```

```
            Delete(line , 1 , Pos(';',line));
            //kry parkeerplek
            bay := line;
             //skep LangTermyn-objek en voeg by skikking by
            pSkik[grootte] := TLangTermyn.Create(reg,entryDT,
exitDT, bay);
          end
          else
          begin
            //skep Daagliks-objek en voeg by skikking by
            pSkik[grootte] := TDaagliks.Create(reg,entryDT, exitDT);
          end;




        end;
    end;
end;
//V4.4 - 4
//opskrif korrek en stuur string terug

function tParkeringBestuurder.naString: string;
var
  i : integer;
  output : string;
begin
   output := '';
   //lus deur skikking
  for i := 1 to grootte do
    begin
      //las by string aan
      //voeg oop reël by
      output := output + pSkik[i].naString() + #13#10 + #13#10;
    end;

  Result := output;
end;

//V4.5 - 6
procedure tParkeringBestuurder.sorteerVolgensRegistrasie;
var
  i , j : integer;
  temp : TDaagliks;
begin
  //buitenste for-lus korrek
  for i := 1 to grootte do
  begin
    //binneste for-lus korrek
    for j := 1 to grootte-1 do
      begin
        //vergelyk die korrekte elemente
        //deur registrasienommer te gebruik
```

```
        if CompareStr(pSkik[j].kryRegistrasieNommer() ,
pSkik[j+1].kryRegistrasieNommer()) > 0 then
        begin
             //stoor in temp met tipe korrek vir skikking
             temp := pSkik[j];
             //ruil elemente om
             pSkik[j] := pSkik[j+1];
             pSkik[j+1] := temp;
        end;

      end;
  end;
end;

//V6.1 - 6
//korrekte metodeopskrif en terugsending
function tParkeringBestuurder.kryRegistrasieLys: string;
var
  i : integer;
  temp : string;
begin
     temp := '';
       //lus deur skikking
     for i := 1 to grootte do
       begin
        //kontroleer of motorregistrasie
        //reeds in temp is
         if NOT( temp.Contains(pSkik[i].kryRegistrasieNommer) ) then
         begin
          //voeg registrasie by lys by
          //voeg 'n #-karakter by
           temp := temp + pSkik[i].kryRegistrasieNommer + '#';
         end;
       end;

     Result:=temp;
end;
//V6.2 - 1
//metodeopskrif en terugsending korrek
function tParkeringBestuurder.genereerRekening: string;
var
  motors, bay, output , currentReg : string;
  temp : TLangTermyn;
  totaal : real;
  i : integer;
  Daagliks : boolean;

begin
  //V6.2.1 - 1
    //roep kryRegistrasieLys en ken aan motors toe
    motors := kryRegistrasieLys();
    bay := '';
    output := '';
```

```
        //isoleer registrasienommers - 3
        //maak skandeerder oop

        //kontroleer vir nog registrasienommers
    while motors.Length > 0 do
    begin
    //onttrek registrasienommer
      currentReg :=  Copy(motors , 1 , Pos('#', motors) - 1);
      Delete(motors , 1 , Pos('#',motors));
      WriteLn(currentReg);
      WriteLn(currentReg.Length);
        //skep 'n string - 3*
       //las die registrasienommer voor die inskrywings aan
      output := output + currentReg  + #13#10;

       //soek en voeg by totaal by - 5
      //inisialiseer totaal binne die lus
      totaal := 0;
      Daagliks := true;
      //lus deur skikking
      for i := 1 to grootte do
      begin
            //kontroleer of motorregistrasie reeds in lys is
          if  ( CompareStr(pSkik[i].kryRegistrasieNommer() ,
                          currentReg) = 0 ) then
          begin
              //voeg parkeergeld by
              //totaal by
              totaal := totaal + pSkik[i].kryParkeerGeld();

              //las elke inskrywing se besonderhede by afvoer aan
              output := output  +
DateTimeToStr(pSkik[i].kryAankomsDatumTyd()) + ' ' +
DateTimeToStr(pSkik[i].kryVertrekDatumTyd()) + ' ' +
FloatToStr(pSkik[i].kryParkeerGeld()) + #13#10;

               //kontroleer of langtermynparkeerplek - 4
               //kontroleer of pSkik[i]
               //'n LangTermyn-objek is
              if pSkik[i] is TLangTermyn then
              begin
                Daagliks := false;
                //ken pSkik[i] aan 'n LangTermyn-objek toe
                temp := pSkik[i] as TLangTermyn;
                //teken parkeerpleknommer aan
                bay := temp.kryParkeerPlek();
              end
              else
              begin
                Daagliks := true;
                bay := '';
              end;
```

```
          end;
      end;
        //las die parkeergeld aan
      output := output + 'Totale parkeergeld verskuldig ' +
FloatToStr(totaal)  + #13#10;
       //voeg parkeerpleknommer by - 2
       //kontroleer of die registrasie 'n LangTermyn-objek is
      if NOT(Daagliks) then
      begin
       //las die parkeerpleknommer aan
          output :=  output  + Nommer van langtermynparkeerplek: '
+ bay  + #13#10 + #13#10
      end
      else
      //voeg boodskap by - 2
      //kontroleer of die totaal bo 1500 is
        if (totaal > 1500 ) AND (Daagliks = true) then
        begin
        //las die boodskap aan
          output := output +  ' Oorweeg dit om langtermynparkering
te huur ' + #13#10 + #13#10;
        end;
     Result:= output;





    end;


end;





end.
```

VRAAG 5, 6.3 ParkeringGK.pas

```
//V5.1 - 1
//klasopskrif korrek
program ParkeringGK;

{$APPTYPE CONSOLE}

{$R *.res}

uses
  System.SysUtils,
  DateUtils,
  uDaagliks in 'uDaagliks.pas',
  uLangTermyn in 'uLangTermyn.pas',
  uParkeringBestuurder in 'uParkeringBestuurder.pas';

var
  pb : tParkeringBestuurder;
begin
  try
    { TODO -oUser -cConsole Main : Insert code here }
    //V5.2 - 2
    //ParkeringBestuurder-objek geskep
    //op korrekte plek
    pb := tParkeringBestuurder.Create();
    //V5.3 - 2
    //sorteermetode geroep
    pb.sorteerVolgensRegistrasie();
     //naString-metode geroep
    WriteLn(pb.naString());

   //V6.3 - 2
   //korrekte metodenaam
   //in 'n afvoerstelling (getipeerde metoderoep)
    WriteLn(pb.genereerRekening());



    ReadLn;
  except
    on E: Exception do
      Writeln(E.ClassName, ': ', E.Message);
  end;
end.
```