|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Opleiding**  **Titel Toetseenheid**  **Titel Toets**  **Code**  **Toetsperiode/studiejaar**  **Maximale score** | | | HBO-ICT  Programmeren Gevorderd  Programmeren Gevorderd Toets  TE.ICT.Vt.P.17.1.PRGV  P3 2017/2018  90 punten | | **Toetsvorm**  **Datum**  **Examinator**  **Nakijker**  **Tijdsduur**  **Cijfer** | | Tentamen  -  Jos Foppele  2 uur  (#punten+10)/100 | |
| **Toegestane hulpmiddelen** | | | Aantekeningen, Presentatie | | | | | |
|  | | | | | | | | |
| Naam |  | | | | | Cijfer | |  |
|  |  |  | | Klas |  |
| Studentnummer | |
|  | |  | |  |  |  | |  |
|  | | | | | | | | |
| **1. Docenten zijn net mensen** | | | | | | | (15 punten) | |
| public class Docent : Mens  {  //class variabelen  private List<Mens> bekenden = new List<Mens>();    //constructor  public Docent(string naam, DateTime datum) {  this.naam = naam;  geboorteDatum = datum;  }  public void AddBekende(Mens m) {  bekenden.Add(m);  }  public void AllemaalEenOnvoldoende() {  foreach (Mens m in bekenden)  if (m is Student)  ((Student)m).OntvangCijfer(5.4);  }  public int GetAantalBekenden() {  return bekenden.Count;  }  } | | | | | | | | |
| **2. Docenten kunnen de boom in** | | | | | | | (10+20 punten) | |
| public void Add(Docent d) {  if (d.GetGeboorteDatum() > this.d.GetGeboorteDatum())  if (jonger == null)  jonger = new TreeD(d);  else  jonger.Add(d);  else  if (ouder == null)  ouder = new TreeD(d);  else  ouder.Add(d);  } | | | | | | | | |
| public void EducatorUpdate() {  DateTime dt = DateTime.Now.AddYears(-35);  if (d.GetGeboorteDatum() > dt) {  d.AllemaalEenOnvoldoende();  if (ouder != null)  ouder.Educatorupdate();  }  if (jonger != null)  jonger.EducatorUpdate();  } | | | | | | | | |
| **3. Studenten met vakken** | | | | | | | (15+15 punten) | |
| public class Vak {  private Docent docent;  private Studenten studenten;  public Vak(Docent d) {  docent = d;  studenten = null;  }  public void Add(Student s) {  if (studenten == null)  studenten = new Studenten(s);  else  studenten.Add(s);  }  public void PrintCijfers() {  if (studenten != null)  studenten.PrintCijfers();  }  } | | | | | | | | |
| public class Studenten {  private Student s;  private Studenten next;  public Studenten(Student s) {  this.s = s;  next = null  }  public void Add(Student s) {  if (next == null)  next = new Studenten(s);  else  next.Add(s);  }  public void PrintCijfers() {  s.PrintCijfers();  if (next != null)  next.PrintCijfers();  }  } | | | | | | | | |
| **4. Pi berekenen** | | | | | | | (15 punten) | |
| public double PiDecimalen(int n) {  double deler = (n\*2.0)\*((n\*2)+1)\*((n\*2)+2);  if (n == 1)  return 4.0/deler;  else  if (n % 2 == 0)  return PiDecimalen(n-1) – (4.0 / deler);  else  return PiDecimalen(n-1) + (4.0 / deler);  } | | | | | | | | |
|  | | | | | | | | |
| **Extra ruimte** | | | | | | | | |
|  | | | | | | | | |