Mecklenburg-Vorpommern



**Zentralabitur 2012**

**Informatik**

**Hinweise für den Lehrer**

**(nicht für die Hand des Prüflings)**

1. Hinweise zur Bewertung

* Die Bewertungshinweise enthalten teilweise keine vollständigen Lösungen, sondern nur Angaben und mögliche Beispielantworten zur erwarteten Schülerleistung. Nicht genannte, aber gleichwertige Lösungswege und Begründungsansätze sind gleichberechtigt.
* Es werden nur ganze Punkte erteilt. Die Vergabe von Zusatzpunkten ist ausgeschlossen.
* Auf Beschluss des Fachprüfungsausschusses können innerhalb einer Aufgabe die Bewertungseinheiten im jeweiligen Anforderungsbereich variiert werden.
* Zur Ermittlung der Gesamtnote werden folgende Tabellen zugrunde gelegt.

Erhöhtes Anforderungsniveau (Hauptfach)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Bewertungseinheiten (BE)** | | | **Notenpunkte** | **Bewertungseinheiten (BE)** | | | **Notenpunkte** |
| 76 | bis | 80 | 15 | 45 | bis | 47 | 07 |
| 72 | bis | 75 | 14 | 41 | bis | 44 | 06 |
| 68 | bis | 71 | 13 | 37 | bis | 40 | 05 |
| 64 | bis | 67 | 12 | 29 | bis | 36 | 04 |
| 60 | bis | 63 | 11 | 22 | bis | 28 | 03 |
| 56 | bis | 59 | 10 | 15 | bis | 21 | 02 |
| 52 | bis | 55 | 09 | 8 | bis | 14 | 01 |
| 48 | bis | 51 | 08 | 0 | bis | 7 | 00 |

Grundlegende Anforderungen (Fach)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Bewertungseinheiten (BE)** | | | **Notenpunkte** | **Bewertungseinheiten (BE)** | | | **Notenpunkte** |
| 57 | bis | 60 | 15 | 34 | bis | 35 | 07 |
| 54 | bis | 56 | 14 | 31 | bis | 33 | 06 |
| 51 | bis | 53 | 13 | 28 | bis | 30 | 05 |
| 48 | bis | 50 | 12 | 22 | bis | 27 | 04 |
| 45 | bis | 47 | 11 | 17 | bis | 21 | 03 |
| 42 | bis | 44 | 10 | 11 | bis | 16 | 02 |
| 39 | bis | 41 | 09 | 6 | bis | 10 | 01 |
| 36 | bis | 38 | 08 | 0 | bis | 5 | 00 |

1. Sonstiges

* Mit den korrigierten Klausuren sind die gedruckten Quelltexte sowie die Datenträger als Anlage aufzubewahren.

Prüfungsteil A

# Olympische Spiele London 2012

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1.1 | Datenbank „Olympische Spiele“ | I | II | III |
| 1.1.1 | Anforderungsanalyse  Konzeptioneller Entwurf formale Beschreibung des Fachproblems, ER-Diagramme  Logischer Entwurf Übertragung des konzeptionellen Entwurfs auf ein konkretes Datenmodell  Physischer Entwurf Tabellen werden implementiert, Zugriffsrechte vergeben, Parameter des DBMS | 2 | 2 | 0 |
| 1.1.2 | Gemeinsamkeiten Zusammenfassung gleichartiger Objekte besitzen Eigenschaften  durch Beziehungen miteinander verbunden  Unterschied Klassen haben Methoden | 1 | 2 | 0 |
| 1.1.3 | 1:n Beziehung  An einem Wettkampfort können mehrere Ereignisse stattfinden. Ein Ereignis findet nur an einem Wettkampfort statt. | 1 | 2 | 0 |
| 1.1.4 | Ereignis, Kategorie  zusammengesetzter Schlüssel bestimmt die Ticketart eindeutig | 0 | 2 | 0 |
| 1.2 | Anfragen an die Datenbank |  |  |  |
| 1.2.1 a) | SELECT WETTKAMPFORT.Name  FROM WETTKAMPFORT  WHERE WETTKAMPFORT.Zone="River Zone"  ORDER BY WETTKAMPFORT.Name | 1 | 1 | 0 |
| 1.2.1 b) | SELECT DISZIPLIN.Name  FROM WETTKAMPFORT INNER JOIN DISZIPLIN   ON WETTKAMPFORT.WID= DISZIPLIN.DID  WHERE WETTKAMPFORT.Zone="Olympiastadion" | 0 | 2 | 1 |
| 1.2.2 | Gib den Wettkampfort für das am 29.07.2012 stattfindende Tennisspiel an. | 1 | 1 | 0 |
| 1.3 | Olympische Wettkampfstätten |  |  |  |
| 1.3.1 | Methoden mit und ohne Parameterübergabe  Methoden mit Rückgabewert  Konstruktor  Bezug zum Beispiel | 1 | 2 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1.3.2 | public void markiereSpielfeldhaelfte()  {  markierTurtle.gehe(118.9);  markierTurtle.dreheRechts(90);  markierTurtle.gehe(109.7);  markierTurtle.dreheRechts(90);  markierTurtle.gehe(118.9);  markierTurtle.dreheRechts(90);  markierTurtle.gehe(109.7);  markierTurtle.dreheRechts(180);  markierTurtle.gehe(13.7);  markierTurtle.dreheRechts(270);  markierTurtle.gehe(118.9);  markierTurtle.dreheRechts(90);  markierTurtle.gehe(82.3);  markierTurtle.dreheRechts(90);  markierTurtle.gehe(118.9);  markierTurtle.dreheRechts(180);  markierTurtle.gehe(54.9);  markierTurtle.dreheRechts(270);  markierTurtle.gehe(82.3);  markierTurtle.dreheRechts(180);  markierTurtle.gehe(41.15);  markierTurtle.dreheRechts(270);  markierTurtle.gehe(64.0);  markierTurtle.dreheRechts(270);  markierTurtle.gehe(54.85);  } | 0 | 2 | 3 |
| 1.3.3 | public void markiereSpielfeld( ) {  this.markiereSpielfeldhaelfte();  markierTurtle.dreheRechts(90);  markierTurtle.gehe(118.9);  markierTurtle.dreheRechts(90);  markierTurtle.gehe(109.7);  markierTurtle.dreheRechts(90);  this.markiereSpielfeldhaelfte();  } | 0 | 1 | 1 |
| 1.3.4 | Verringerung des Kreideverbrauch durch Anheben des Stiftes beim Überfahren bereits gezeichneter Linien | 0 | 1 | 0 |
|  | **gesamt** | 7 | 18 | 5 |

# Call a Bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2.1 | Modellierung | I | II | III |
| 2.1.1 | Ein Datenbanksystem (DBS) besteht aus einer Datenbasis und einem Datenbank-Managementsystem (DBMS).  Das DBMS stellt die Schnittstelle zwischen Benutzer und Datenbasis dar und dient der effizienten Speicherung und Abfrage der strukturierten Daten.  Die Datenbasis beinhaltet alle notwendigen Daten zur Realisierung der gewünschten Anforderungen. | 3 | 0 | 0 |
| 2.1.2 | n:1 (Ein Fahrrad kann zu einem bestimmten Zeitpunkt nur an einem Ort abgestellt werden. An einem Ort können mehrere Fahrräder zur selben Zeit stehen.) | 0 | 2 | 0 |
| 2.1.3 | Einsatzbereitschaft: Datentyp Boolean (Es gibt genau zwei Möglichkeiten entweder ist das Fahrrad einsatzbereit oder nicht einsatzbereit)  Code: Datentyp Integer oder Text (Der Typ Integer reicht aus, um eine genügend große Anzahl von Kombinationen bereitzustellen und eine ausreichende Sicherheit zu gewährleisten. Führende Nullen sind dabei ausgeschlossen und können nur über den Datentyp Text realisiert werden.) | 1 | 2 | 0 |
| 2.1.4 | KUNDE(**Kundennummer**, Name, Vorname, Telefonnummer, PLZ, Ort, Strasse, Hausnummer)  MIETET(***Kundennummer***, ***Fahrradnummer***, **vonDatum**, bisDatum, BeginnZeit, EndeZeit)  FAHRRAD(**Fahrradnummer**, Entsperrcode, Einsatzbereitschaft, *Standortnummer*)  STANDORT(**Standortnummer**, PLZ, Ort, Strasse, Hausnummer) | 1 | 1 | 2 |
| 2.1.5 | Kennwortauthorisierung, verteilte Datenspeicherung, Trennung von Ausleih- und Bezahlvorgang, keine Vorratsdatenspeicherung, keine Erstellung von Bewegungsprofilen einschließlich Beschreibung | 0 | 2 | 2 |
| **2.2** | **SQL-Abfragen** |  |  |  |
| 2.2.1 | SELECT \* FROM FAHRRAD  WHERE FAHRRAD.Einsatzbereitschaft = false; | 0 | 1 | 0 |
| 2.2.2 | SELECT KUNDE.Name  FROM MIETET InnerJOIN KUNDE  ON KUNDE.Kundennummer = MIETET.Kundennummer  WHERE MIETET.bisDatum = "14.05.2012"; | 0 | 1 | 1 |
| 2.3 | Mobilfunk und Co |  |  |  |
| 2.3.1 | Gemeinsamkeit:  Übertragung von Daten  Unterschiede:  persönlicher Kontakt beim Telefonieren  Zugangsdaten zum Bahnhofshotspot erforderlich | 0 | 1 | 2 |
| 2.3.2 | Laptop mit WLAN-Adapter, WLAN-fähiges Mobiltelefon | 2 | 0 | 0 |
| 2.3.3 | Beschreibung des Datenflusses mit Hilfe des OSI- bzw. des TCP/IP-Schichtenmodells  Netzzugangsschicht: Empfang der Funkdaten  Internetschicht: Auflösung der IP-Adresse  Transportschicht: Vollständigkeit der Pakete, Reihenfolge, Korrektheit der Übertragung, Zuordnung des Anwendungsports  Anwendungsschicht: Anzeige der Webseite im Browser | 1 | 2 | 0 |
| 2.3.4  a) | 14,4 MBit \* 60 / 8 = 108 MByte | 0 | 2 | 0 |
| 2.3.4  b) | Verkleinerung der Bandbreite bei mehreren aktiven Nutzern | 0 | 1 | 0 |
|  | **gesamt** | 8 | 15 | 7 |

# IBAN – International Bank Account Number

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **3.1** | **Analyse der IBAN** | **I** | **II** | **III** |
| 3.1.1 | Zeichenkette  Zu speichernde Werte bestehen aus Buchstaben und Ziffern. Dies kann kein anderer Datentyp außer Zeichenkette leisten. | **1** | **1** | **0** |
| 3.1.2 | 700519950000007229 | **1** | **0** | **0** |
| **3.2** | **IBAN-Leser als Akzeptor** |  |  |  |
| 3.2.1 | Da es sich um eine IBAN handelt, muss der Schüler die Länge der IBAN und die Gültigkeit der Prüfzahl nicht testen. | **0** | **2** | **1** |
| 3.2.2 | Eingabealphabet {0..9, A..Z}, Zustandsmenge {z0, …, z8}  Automat liest das aktuelle Zeichen und wechselt in Abhängigkeit vom gegenwärtigen Zustand in den Folgezustand  Wird nach dem Lesen des letzten Zeichens der Endzustand z8 erreicht, handelt es sich um eine gültige IBAN.  Bezug zum Beispiel | **1** | **2** | **0** |
| **3.3** | **Berechnung** **der Prüfzahl** |  |  |  |
| 3.3.1 |  | **1** | **1** | **0** |
| 3.3.2 | Alle Attribute sind private definiert. Sie stehen der Klasse IBANGenerator uneingeschränkt zur Verfügung. Der Zugriff auf diese Attribute von anderen Klassen heraus ist nur mit speziellen Methoden möglich.  So kann das Attribut iban mit Hilfe der als public deklarierten Methode getIBAN gelesen werden.  Das Attribut pruefzahl kann nur innerhalb der Klasse verwendet werden, da kein spezieller Leseauftrag implementiert ist. | **1** | **2** | **0** |
| 3.3.3 | Die Aufgabe übernimmt die Methode setKontonummer.  Besteht die Kontonummer aus mehr als 10 Ziffern, wird die Fehlermeldung: "Fehler: Kontonummer ist länger als 10 Ziffern!" ausgegeben und die Kontonummer erhält den Wert null.  Bei anderen ungültigen Eingaben wird die Fehlermeldung: "Fehler: Kontonummer ist ungültig!" ausgegeben und die Kontonummer erhält auch den Wert null. | **0** | **2** | **0** |
| 3.3.4 a) | int laendercodezeichen1 = ((int) laendercode.charAt(0)) - 55;  int laendercodezeichen2 = ((int) laendercode.charAt(1)) - 55;  Die Buchstaben des Ländercodes werden der entsprechenden Zahl im ASCII-Code zugeordnet: A = 65, B = 66, … Die anschließende Subtraktion von 55 führt zu der Zuordnung: A = 10, B = 11, … | **0** | **0** | **2** |
| 3.3.4 b) | return bban + laendercodezeichen1 + laendercodezeichen2 + "00";  Die modifizierte IBAN besteht aus BBAN, numerischem Ländercode und 00 am Ende in genannter Reihenfolge. | **0** | **2** | **0** |
| 3.3.5 | private void korrigierePruefzahl(int pWert)  {  int wert = pWert;  wert = 98 - wert;  if (wert < 10)  {  pruefzahl = "0" + wert;  }  else  {  pruefzahl = "" + wert;  }  } | **0** | **2** | **2** |
| 3.3.6 | public void erzeugeIBAN()  {  bban = bankleitzahl + kontonummer;  String modifizierteIBAN = this.getModifizierteIBAN();  erzeugePruefzahl(modifizierteIBAN);  iban = laendercode + pruefzahl + bban;  } | **0** | **2** | **1** |
| 3.3.7 | public String getIBAN()  {return iban;} | **1** | **0** | **0** |
| 3.3.8 | Beschreibung von zwei Vorteilen, z. B.   * strikte Trennung von Benutzeroberfläche (GUI) und Fachklasse (IBANGenerator) erleichtert Aktualisierungen * Arbeit im Team, Implementierung einzelner Klassen kann auf verschiedene Personen aufgeteilt werden | **0** | **2** | **0** |
|  | **gesamt** | **6** | **18** | **6** |

# Netzwerkfähiger Plotter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4.1 | Von-Neumann-Architektur | I | II | III |
| 4.1.1 |  | 2 | 0 | 0 |
| 4.1.2 | Aufbau identisch: Speicherzellen gleicher Größe fortlaufend angeordnet; bei Von-Neumann: durch Adresse nummeriert  Funktionsweise Kellerspeicher: Zugriff auf das oberste Elemenent mit POP, Auflegen eines Elements mit PUSH(Element) Funktionsweise Von-Neumann: Zugriff über Adresse auf jedes Element jederzeit möglich (wahlfreier Zugriff) | 0 | 3 | 0 |
| 4.1.3 | FETCH-Phase/Befehlshole-Phase  Aus dem Speicher wird der nächste, abzuarbeitende Befehl in das Befehlsregister geladen. Die zu benutzende Speicheradresse gibt der Befehlszähler an.  DECODE-Phase/Dekodier-Phase  Der Befehlszähler wird um Eins erhöht und der Befehl im Steuerwerk übersetzt/dekodiert.  EXECUTE-Phase/Ausführungs-Phase  Gegebenenfalls werden aus dem Speicher Operanden geholt (FETCH OPERANDS). Der Befehl wird im Rechenwerk ausgeführt. Im Falle eines Sprungbefehls wird auch der Befehlszähler verändert. | 0 | 4 | 0 |
| 4.2 | Netzanbindung |  |  |  |
| 4.2.1 | Subnetzmaske in beiden Netzen 255.255.255.0, Gruppe 1 ist demnach im Netz 192.168.1.0 und der Plotter im Netz 192.168.3.0 | 1 | 0 | 0 |
| 4.2.2 | SWITCH arbeitet auf OSI Layer 2 (Netzverbund im gleichen Segment); wenn alle Zugriff auf den Plotter haben sollen, müssen die Geräte dann im gleichen Netz sein 🡪 Widerspruch zur Sicherheitsforderung | 0 | 0 | 1 |
| 4.2.3 | Einsatz eines Routers mit festgelegten Routen zum Plotter und Routingsperre zwischen den Netzen der beiden Gruppen oder  Nutzung beider Plotteranschlüsse  Netzwerkanschluss – für Forschergruppe 1 mit einer Adresse aus dem 192.168.1.0/24-Netz  USB-Anschluss an Printserver anschließen und diesen als Netzgerät im Netz 192.168.2.0/24 konfigurieren. | 0 | 1 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4.3 | Plottersprache HPGL |  |  |  |
| 4.3.1 |  | 1 | 0 | 0 |
| 4.3.2 | Der erste Befehl ist fehlerhaft, da dieser eine Dezimalzahl als zweiten Parameter verwendet. Dies ist nach Syntaxdiagramm KOORDINATEN nicht zulässig. Die anderen Befehle sind korrekt. | 0 | 2 | 0 |
| 4.3.3 | Syntaxdiagramm2.wmf | 0 | 1 | 0 |
| 4.3.4 | Typ 2 – kontextfreie Grammatik, da:  Typ 0 – Grammatik,  Typ 1 – Die rechte Seite ist stets länger als die linke Seite.  Typ 2 – Auf der linken Seite steht immer genau ein Nichtterminalsymbol (Bezeichner am jeweiligen Syntaxdiagramm).  Typ 3 – Liegt nicht vor. Auf der rechten Seite steht z. B.  <HPGL> 🡪 <Befehl> ";" | <Befehl> ";" <HPGL>. | 0 | 1 | 1 |
| 4.3.5 | Kellerautomat oder Turing-Maschine | 1 | 0 | 0 |
|  | **Gesamt BE** | 5 | 12 | 3 |

# 

# Twitter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5.1 | Twittermodell | I | II | III |
| 5.1.1 | Die Tabelle enthält z. B. keine Angaben zum Following. | 1 | 0 | 0 |
| 5.1.2 | Hashtags dienen zur Kennzeichnung von Schlagwörtern. Sie werden durch das Zeichen # eingeleitet und mit einem Leerzeichen beendet.  Hashtags in der Tabelle sind Botschaft, Dorf, Kaffee und Fussballliga. | 2 | 0 | 0 |
| 5.2 | Normalisierung von Twitterdaten |  |  |  |
| 5.2.1 | Die Tabelle speichert die Angaben zu den Nutzern redundant. Das kann Anomalien hervorrufen, z. B. eine Änderungsanomalie bei der zeilenweisen Überarbeitung der E-Mail-Adresse von flor94. | 0 | 2 | 0 |
| 5.2.2 | Der Primärschlüssel ist eine minimale Attributkombination zur Unterscheidung aller Datensätze in der Tabelle. Da die E-Mail-Adressen der Nutzer eindeutig sind und das System zu einem Zeitpunkt nur einen Tweet registrieren kann, ist die angegebene Attributkombination ausreichend und sinnvoll. | 1 | 1 | 0 |
| 5.2.3 | Überführung in die 3NF in Abhängigkeit des gewählten Primärschlüssels inklusive der Kommentierung und Darstellung der ausgeführten Schritte:  BENUTZER(Benutzer, E-Mail, Benutzername)  TWEETS(*E-Mail*, Absender, AbsenderZeit, AbsenderOrt, Tweet)  (Primärschlüssel, *Fremdschlüssel*)  Auf eine Atomarisierung der Benutzer und AbsenderZeit kann verzichtet werden, da sie im Sinne der Aufgabe atomar sind. | 1 | 3 | 1 |
| 5.3 | Trending Topics |  |  |  |
| 5.3.1 | Das Attribut tag ist als Liste implementiert. Dadurch ist die Anzahl der gespeicherten Hashtags flexibel. | 0 | 2 | 0 |
| 5.3.2 | Solange im tweet noch ein #-Zeichen vorhanden ist, wird zunächst der Text bis einschließlich des #-Zeichens gelöscht.  Nun wird ein Leerzeichen gesucht. Falls dieses existiert, wird der Text bis zum Leerzeichen an die tag -Liste angehängt und aus dem tweet gelöscht.  Wenn kein Leerzeichen existiert, ist der verbleibende tweet ein Hashtag und wird an die tag -Liste angefügt, falls er nicht leer ist. | 0 | 2 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5.3.3 | public String bestimmeAnzahl(String pText)  {  String text = pText;  int anzahl = 0;  for (String element: tag)  {  if (text.equals(element)) {anzahl++;}  }  return ""+anzahl;  } | 0 | 1 | 1 |
| 5.3.4 | public ArrayList<String> gibBereinigteListe()  {  ArrayList<String> bereinigteListe = new ArrayList<String>();  bereinigteListe.clear();  for (String element: tag)  {  if (!bereinigteListe.contains(element))  {  bereinigteListe.add(element);  }  }  return bereinigteListe;  } | 0 | 1 | 1 |
|  | gesamt | 5 | 12 | 3 |