abitur.nrw
BERUFSKOLLEG
Berufliches Gymnasium

Vorgaben für die Abiturprüfung 2027

in den Bildungsgängen des Beruflichen Gymnasiums
Anlagen D 1 – D 28
Profil bildendes Leistungskursfach

Informatik

Fachbereich Informatik

1 Gültigkeitsbereich

Die Vorgaben für die Abiturprüfung im Fach Informatik gelten für folgenden Bildungsgang:

Allgemeine Hochschulreife (Mathematik, Informatik)	APO-BK, Anlage D 21

Der Bildungsgang ist dem Fachbereich Informatik zugeordnet.

2 Vorgaben für die schriftliche Abiturprüfung

Grundlage für die Vorgaben der zentral gestellten schriftlichen Aufgaben der Abiturprüfung der (mindestens) dreijährigen AHR-Bildungsgänge des Beruflichen Gymnasiums (APO-BK, Anlagen D 1 – D 28) sind die verbindlichen Vorgaben der Bildungspläne zur Erprobung (RdErl. d. Ministeriums für Schule und Weiterbildung des Landes Nordrhein-Westfalen v. 30.6.2006):

Teil I: Pädagogische Leitideen,

Teil II: Didaktische Organisation der Bildungsgänge im Fachbereich Informatik,

Teil III: Fachlehrplan Informatik.

Durch die Vorgaben für die schriftliche Abiturprüfung werden inhaltliche Schwerpunkte festgelegt. Diese inhaltlichen Schwerpunkte sind Konkretisierungen der in dem Fachlehrplan beschriebenen Fachinhalte, deren Behandlung im Unterricht als Vorbereitung auf die schriftliche Abiturprüfung vorausgesetzt wird. Durch diese Schwerpunktsetzungen soll sichergestellt werden, dass alle Schülerinnen und Schüler, die im Jahr 2027 das Abitur in dem o. a. Bildungsgang des Beruflichen Gymnasiums ablegen, über die Voraussetzungen zur Bearbeitung der zentral gestellten Aufgaben verfügen.

Die folgenden fachspezifischen Schwerpunktsetzungen gelten für das Jahr 2027. Sie stellen keine dauerhaften Festlegungen dar.

3 Verbindliche Unterrichtsinhalte im Fach Informatik im Fachbereich Informatik für das Abitur 2027

3.1 Inhaltliche Schwerpunkte

Datenbanksysteme

- Ausgangspunkt: durchgängige Handlungssituation
- Grundlagen von Datenbanken, Aufgaben eines DBMS, Architektur eines Datenbanksystems (Drei-Ebenen-Architektur)
- Erstellung eines ER-Modells mittels des Programms „Dia“ in der MC-Notation mit in der Regel vier bis zwölf Entitäten einschließlich Kardinalitäten, Umsetzung eines ER-Modells in ein relationales Datenbankmodell
- Erstellung eines relationalen Datenbankmodells
- Normalisierung bis einschließlich 3. Normalform
- SQL-Datenbanksprache:
    - DDL mit Erzeugen, Ändern und Löschen von Tabellen, dabei auch Erstellen und Löschen von Schlüsseln und referenzielle Integrität, Erstellen und Löschen von Sichten (Views)
    - DML mit Update-, Insert- und Delete-Befehl
    - DQL mit Abfragebefehlen unter Nutzung von Aggregatfunktionen, Gruppierungen, Sortierungen, einstufiger Schachtelung bei Unterabfragen, Left-Join, Right-Join und Inner-Join; Umsetzung anhand von gegebenen Tabellen

Kryptologie

- Schutzziele und Bedrohungen
- symmetrische Verschlüsselung nach dem Vigenère-Verfahren
- Analyse, Modifikation bzw. Variation eines gegebenen symmetrischen Verschlüsselungsverfahrens
- Schlüsselvereinbarung bei symmetrischer Verschlüsselung mit dem Diffie-Hellman-Verfahren (hybride Verfahren)
- asymmetrische Verschlüsselung mit dem RSA-Verfahren, Satz von Euler, Sicherheit des RSA-Verfahrens; modulare Arithmetik, wiederholtes Quadrieren und Multiplizieren, erweiterter Euklidischer Algorithmus, Vielfachsummendarstellung
- Kryptoanalyse
- digitale Signatur; Anforderungen an Hash-Funktionen, Anwendung von Hash-Funktionen bei der digitalen Signatur; Zertifizierung von Schlüsseln
- Realisierung von kryptologischen Algorithmen in Java-Quellcode ggf. unter Nutzung eines Texteditors (ohne Java-Compilernutzung)
- Beschreibung und Begründung der Einsatzmöglichkeiten der oben genannten kryptologischen Verfahren im Rahmen einer Handlungssituation

Dynamische Datenstrukturen

- Ausgangspunkt: durchgängige Handlungssituation
- Konzeption dynamischer Datenstrukturen
- Schlange, Stapel und Lineare Liste (einfach- und doppelt verkettete Liste) von Objekten
- Anwendung der Standardoperationen (vgl. Anhang), Erstellung von Java-Quellcode ggf. unter Nutzung eines Texteditors (ohne Java-Compilernutzung)
- Implementation von dynamischen Datenstrukturen und ihren Operationen in Java-Quellcode ggf. unter Nutzung eines Texteditors (ohne Java-Compilernutzung)

Software-Engineering

- Ausgangspunkt: durchgängige Handlungssituation
- Entwicklung und Erstellung eines Klassendiagramms (Unified Modeling Language UML) mit in der Regel drei bis acht Klassen mittels des Programms „Dia“ mit den Beziehungsarten Assoziation, Aggregation, Komposition, Vererbung (auch abstrakte Klassen und Nutzung von Polymorphie)
- Darstellung dynamischer Prozesse mit Sequenzdiagrammen
- Realisierung von Assoziation, Aggregation, Komposition und Vererbung (auch abstrakte Klassen und Nutzung von Polymorphie) in Java-Quellcode ggf. unter Nutzung eines Texteditors (ohne Java-Compilernutzung)
- Prinzipien und Vorgehensmodelle des Software Engineerings

3.2 Medien/Materialien

keine

3.3 Formale Hinweise

- Entity-Relationship-Modelle (ERM) werden in MC-Notation gegeben oder sind in MC-Notation zu erstellen.
- Klassen- und Sequenzdiagramme sind in Unified Modeling Language (UML) zu erstellen.

3.4 Hinweise zu den Aufgabenstellungen

Die Aufgaben in den zentral gestellten Prüfungen werden mit Hilfe von Operatoren formuliert.

In der folgenden Tabelle werden die Operatoren definiert, durch Beispiele dokumentiert und den Anforderungsbereichen (AFB I, II und III) zugeordnet. Die konkrete Zuordnung erfolgt immer im Kontext der Aufgabenstellung, wobei eine eindeutige Trennung der Anforderungsbereiche nicht immer möglich ist.

Spätestens in der Qualifikationsphase sollen die Operatoren in den Klausuren und schriftlichen Übungen verwendet werden, um die Schülerinnen und Schüler auf die Abiturprüfung vorzubereiten.

Operator	**AFB**	**Definition**	**Beispiel**
angeben	I	Elemente, Sachverhalte, Begriffe, Daten ohne nähere Erläuterungen aufzählen	Geben Sie die Bedingung für die erste Normalform an.
nennen (Synonym angeben)	I	Elemente, Sachverhalte, Begriffe, Daten ohne nähere Erläuterungen aufzählen	Nennen Sie die Schichten des TCP/IP-Modells.
berechnen, bestimmen	I, II	mittels Größenvorgaben eine informatische Größe unter Angabe des Rechenweges ermitteln	Berechnen Sie einen geheimen Schlüssel beim RSA-Verfahren mit den Primzahlen p=13 und q=17 und dem öffentlichen Schlüssel e=5.
beschreiben	I, II	Strukturen, Sachverhalte oder Zusammenhänge strukturiert und fachsprachlich richtig mit eigenen Worten wiedergeben	Beschreiben Sie den Unterschied zwischen einer rekursiven und einer iterativen Vorgehensweise bei der Berechnung des GGT.

Operator	AFB	Definition	Beispiel
darstellen, dokumentieren	I, II	Sachverhalte, Zusammenhänge, Methoden etc. strukturiert und gegebenenfalls fachsprachlich wiedergeben	Stellen Sie dar, welche Aufgaben das TCP-Protokoll und welche Aufgaben das IP-Protokoll übernimmt.
ermitteln	I, II	einen Zusammenhang oder eine Lösung finden und das Ergebnis formulieren	Ermitteln Sie, welche der folgenden IP-Adressen einem Host nicht zugewiesen werden kann.
erklären	I, II	Sachverhalte mit Hilfe eigener Kenntnisse verständlich und nachvollziehbar machen und in Zusammenhänge einordnen	Erklären Sie die Begriffe symmetrische und asymmetrische Verschlüsselung.
konvertieren	I, II	umwandeln von Darstellungen in verschiedene Systeme	Konvertieren Sie die IP-Adressen in das binäre Format.
überführen	I, II	Darstellung in eine andere Darstellungsform bringen	Überführen Sie das gegebene ER-Diagramm in ein Relationenmodell.
zeichnen, skizzieren	I, II	eine anschauliche und hinreichend exakte grafische Darstellung gegebener Strukturen anfertigen	Zeichnen Sie nach den Angaben eine eEPK.
zusammen-fassen	I, II	das Wesentliche in konzentrierter Form herausstellen	Fassen Sie den gegebenen Sachtext im Hinblick auf den Datenschutz zusammen.
erläutern	II	einen Sachverhalt durch zusätzliche Informationen veranschaulichen und verständlich machen	Erläutern Sie den Grund für die Ungültigkeit der IP-Adresse.

Operator	AFB	Definition	Beispiel
analysieren	II, III	Analyse ist eine systematische Untersuchung, bei der der untersuchte Gegenstand in seine Bestandteile zerlegt wird und diese anschließend geordnet, untersucht und ausgewertet werden.	Analysieren Sie den Aufwand, den dieser Algorithmus hinsichtlich der Vergleiche von Feldelementen im schlechtesten Fall benötigt.
ausführen, durchführen	II, III	Durchlaufen eines Prozesses mit Angabe der Ergebnisse	Führen Sie diesen Algorithmus mit Hilfe der nachstehenden Beispielzahlen durch.
begründen	II, III	Sachverhalte auf Gesetzmäßigkeiten bzw. kausale Zusammenhänge zurückführen	Begründen Sie, warum eine buchstabenweise Verschlüsselung keine sichere Verschlüsselung darstellt.
beweisen	II, III	eine Aussage mit Hilfe logischer Schlussfolgerungen bestätigen	Beweisen Sie, dass das Laufzeitverhalten des Bubble Sort im schlechtesten Fall $O(n^2)$ ist.
bewerten, beurteilen	II, III	Sachverhalte, Gegenstände, Methoden, Ergebnisse etc. an Beurteilungskriterien oder Normen und Werten messen	Bewerten Sie anhand von ausgewählten Aspekten Stärken und/oder Schwächen der eEPK.
diskutieren	II, III	in Zusammenhang mit Sachverhalten, Aussagen oder Thesen unterschiedliche Positionen bzw. Pro- und Contra-Argumente einander gegenüber-stellen und abwägen	Diskutieren Sie die Vor- und Nachteile des folgenden ER-Modells zur Lösung des geschilderten betrieblichen Problems.

Operator	AFB	Definition	Beispiel
entwerfen	II, III	Strukturen, Abläufe erfassen; eine Lösungsidee entwickeln und darstellen; Lösungsansatz für ein Problem	Entwerfen Sie einen Algorithmus zur Lösung des geschilderten Problems.
erstellen	II, III	darstellen von Sachverhalten gemäß vorgegebener Syntax	Erstellen Sie aus der obigen Notiz ein Klassendiagramm gemäß Unified Modeling Language.
erweitern, vervollständigen	II, III	einer gegebenen Struktur weitere Bestandteile hinzufügen	Erweitern Sie das gegebene Struktogramm.
implementieren	II, III	in eine Programmiersprache (auch Pseudocode) übersetzen	Implementieren Sie nach gegebenem Struktogramm die Methode erhoeheLohn() für die Klasse Mitarbeiter.
konstruieren	II, III	entwickeln eines Produktes zu vorgegebenen Eigenschaften	Konstruieren Sie ein Beispiel für eine Relation, die die Bedingungen der ersten Normalform, aber nicht die Bedingungen der zweiten Normalform erfüllt.
modellieren	II, III	zu einem Ausschnitt aus der Realität ein informatisches Modell anfertigen.	Modellieren Sie anhand der gemachten Angaben die gewünschte Datenbank in einem ER-Diagramm in MC-Notation.
planen	II, III	zusammenstellen von Funktionalitäten unter Berücksichtigung vorgegebener Daten	Planen Sie ein Netzwerk für ein Unternehmen gemäß der Vorgaben.
überprüfen	II, III	Aktuelles mit Vorgaben vergleichen	Überprüfen Sie, ob das gegebene ER-Modell der Unternehmenssituation entspricht.

Operator	AFB	Definition	Beispiel
untersuchen	II, III	sukzessive Überprüfung von Einzelfällen	Untersuchen Sie in folgendem Szenario das Problem der gegebenen Subnetzmaske.
vergleichen	II, III	gegenüberstellen von Ergebnissen/Objekten, dabei Gemeinsamkeiten, Ähnlichkeiten und Unterschiede ermitteln	Vergleichen Sie die beiden Sortierverfahren im Hinblick auf das Laufzeitverhalten.

4 Arbeitszeit für die schriftliche Abiturprüfung

Es gelten die Vorgaben der APO-BK, § 17 (2) Anlage D.

Die Arbeitszeit beträgt 270 Minuten.

Als Rüstzeit für die Nutzung von Computersystemen wird eine Zeit von 30 Minuten vorgesehen, die außerhalb der Arbeitszeit liegt.

5 Hilfsmittel

- Wörterbuch der deutschen Rechtschreibung
- modulares Mathematiksystem (MMS)
- Eine Nutzung von Computersystemen ist verpflichtend vorgesehen. Als Programme werden „Dia“ Version 0.97 oder höher und ein Texteditor eingesetzt.

Hinweise zur Computernutzung:

- ER-Diagramme und Klassendiagramme müssen mit dem Programm „Dia“ erstellt werden.
- Das Programm „Dia“ unterliegt der GNU General Public License und kann somit ohne weitere Kosten installiert werden.
- Der Texteditor darf nur zur Erstellung von Quellcode verwendet werden. Jegliche Unterstützung einer Programmiersprache durch den Texteditor ist auszuschließen (keine Hervorhebung der Syntax, kein automatisches Einrücken, keine Überprüfung der Syntax etc.)
- Eine systemtechnisch abgeschlossene Prüfungsumgebung ist sicher zu stellen.

Folgende Anforderungen sind zu beachten:

- Der Zugriff durch Prüflinge auf irgendeinen Internet-Dienst ist auszuschließen.
- Der Zugriff durch Prüflinge auf irgendeinen Netzwerkdienst, der nicht für die Aufgabenerledigung notwendig ist, ist auszuschließen.
- Der Zugriff durch Prüflinge auf bestehende Dateien, Texte, Dokumente, die nicht für die Aufgabenerledigung notwendig sind, ist auszuschließen.
- Eine Rechnernutzung für Teilaufgaben, bei denen die Rechnernutzung nicht ausdrücklich als Hilfsmittel zugelassen ist, ist auszuschließen bzw. als Täuschungsversuch zu werten.
- Nach dem Erstellen der Prüfungsleistung ist ein Ausdruck am Prüfungstag unmittelbar nach der eigentlichen Bearbeitungszeit im Rahmen der Rüstzeit durchzuführen. Der Ausdruck ist Teil der Prüfungsleistung und vom Prüfling durchzuführen.
- Aufsichtführende wirken bei der Durchführung des Ausdrucks unterstützend mit. Die Prüfungsleistung ist in ihrem gegebenen Zustand auszudrucken, ohne dass weitere Optimierungen seitens der Aufsichtsführenden durchgeführt werden.
- Der Ausdruck ist durch den Prüfling mit Datumsangabe zu unterschreiben. Fehldrucke sind durch den Prüfling zu entwerten und durch den Prüfling mit Datumsangabe zu unterschreiben.
- Falls die jeweilige Aufgabe gar nicht bearbeitet worden ist, ist ein weißes Blatt mit der Bezeichnung der jeweiligen Aufgabe durch den Prüfling mit Datumsangabe zu unterschreiben.

6 Hinweise zur Aufgabenauswahl durch die Lehrkraft/ den Prüfling

Eine Aufgabenauswahl durch die Schule ist nicht vorgesehen.

Eine Aufgabenauswahl durch die Prüflinge ist ebenfalls nicht vorgesehen.

Anhang

In Java werden Objekte über Referenzen verwaltet, d. h., eine Variable pObject von der Klasse Object enthält eine Referenz auf das entsprechende Objekt. Zur Vereinfachung der Sprechweise werden jedoch im Folgenden die Referenz und das referenzierte Objekt sprachlich nicht unterschieden.

Die Klasse Queue

Objekte der Klasse **Queue** (Warteschlange) verwalten beliebige Objekte nach dem First-In-First-Out-Prinzip, d. h., das zuerst abgelegte Objekt wird als erstes wieder entnommen.

Dokumentation der Klasse Queue

Konstruktor	**Queue()** Eine leere Schlange wird erzeugt.
Anfrage	**boolean isEmpty()** Die Anfrage liefert den Wert true, wenn die Schlange keine Objekte enthält, sonst liefert sie den Wert false.
Auftrag	**void enqueue(Object pObject)** Das Objekt pObject wird an die Schlange angehängt. Falls pObject gleich null ist, bleibt die Schlange unverändert.
Auftrag	**void dequeue()** Das erste Objekt wird aus der Schlange entfernt. Falls die Schlange leer ist, wird sie nicht verändert.
Anfrage	**Object front()** Die Anfrage liefert das erste Objekt der Schlange. Die Schlange bleibt unverändert. Falls die Schlange leer ist, wird null zurückgegeben.

Die Klasse Stack

Objekte der Klasse **Stack** (Keller, Stapel) verwalten beliebige Objekte nach dem Last-In-First-Out-Prinzip, d. h., das zuletzt abgelegte Objekt wird als erstes wieder entnommen.

Dokumentation der Klasse Stack

Konstruktor	**Stack()** Ein leerer Stapel wird erzeugt.
Anfrage	**boolean isEmpty()** Die Anfrage liefert den Wert true, wenn der Stapel keine Objekte enthält, sonst liefert sie den Wert false.
Auftrag	**void push(Object pObject)** Das Objekt pObject wird oben auf den Stapel gelegt. Falls pObject gleich null ist, bleibt der Stapel unverändert.
Auftrag	**void pop()** Das zuletzt eingefügte Objekt wird von dem Stapel entfernt. Falls der Stapel leer ist, bleibt er unverändert.
Anfrage	**Object top()** Die Anfrage liefert das oberste Stapelobjekt. Der Stapel bleibt unverändert. Falls der Stapel leer ist, wird null zurückgegeben.

Die Klasse List

Objekte der Klasse **List** verwalten beliebig viele, linear angeordnete Objekte. Auf höchstens ein Listenobjekt, aktuelles Objekt genannt, kann jeweils zugegriffen werden. Wenn eine Liste leer ist, vollständig durchlaufen wurde oder das aktuelle Objekt am Ende der Liste gelöscht wurde, gibt es kein aktuelles Objekt. Das erste oder das letzte Objekt einer Liste können durch einen Auftrag zum aktuellen Objekt gemacht werden. Außerdem kann das dem aktuellen Objekt folgende Listenobjekt zum neuen aktuellen Objekt werden.

Das aktuelle Objekt kann gelesen, verändert oder gelöscht werden. Außerdem kann vor dem aktuellen Objekt ein Listenobjekt eingefügt werden.

Dokumentation der Klasse List

Konstruktor	**List()** Eine leere Liste wird erzeugt.
Anfrage	**boolean isEmpty()** Die Anfrage liefert den Wert true, wenn die Liste keine Objekte enthält, sonst liefert sie den Wert false.
Anfrage	**boolean hasAccess()** Die Anfrage liefert den Wert true, wenn es ein aktuelles Objekt gibt, sonst liefert sie den Wert false.
Auftrag	**void next()** Falls die Liste nicht leer ist, es ein aktuelles Objekt gibt und dieses nicht das letzte Objekt der Liste ist, wird das dem aktuellen Objekt in der Liste folgende Objekt zum aktuellen Objekt, andernfalls gibt es nach Ausführung des Auftrags kein aktuelles Objekt.
Auftrag	**void toFirst()** Falls die Liste nicht leer ist, wird das erste Objekt der Liste aktuelles Objekt. Ist die Liste leer, geschieht nichts.
Auftrag	**void toLast()** Falls die Liste nicht leer ist, wird das letzte Objekt der Liste aktuelles Objekt. Ist die Liste leer, geschieht nichts.
Anfrage	**Object getObject()** Falls es ein aktuelles Objekt gibt, wird das aktuelle Objekt zurückgegeben, andernfalls gibt die Anfrage den Wert null zurück.
Auftrag	**void setObject(Object pObject)** Falls es ein aktuelles Objekt gibt und pObject ungleich null ist, wird das aktuelle Objekt durch pObject ersetzt.

Auftrag	**void append(Object pObject)**
	Ein neues Objekt pObject wird am Ende der Liste angefügt. Das aktuelle Objekt bleibt unverändert. Wenn die Liste leer ist, wird das Objekt pObject in die Liste eingefügt und es gibt weiterhin kein aktuelles Objekt. Falls pObject gleich null ist, bleibt die Liste unverändert.
Auftrag	**void insert(Object pObject)**
	Falls es ein aktuelles Objekt gibt, wird ein neues Objekt vor dem aktuellen Objekt in die Liste eingefügt. Das aktuelle Objekt bleibt unverändert. Falls die Liste leer ist und es somit kein aktuelles Objekt gibt, wird pObject in die Liste eingefügt und es gibt weiterhin kein aktuelles Objekt. Falls es kein aktuelles Objekt gibt und die Liste nicht leer ist oder pObject gleich null ist, bleibt die Liste unverändert.
Auftrag	**void concat(List pList)**
	Die Liste pList wird an die Liste angehängt. Das aktuelle Objekt bleibt unverändert. Falls pList null oder eine leere Liste ist, bleibt die Liste unverändert.
Auftrag	**void remove()**
	Falls es ein aktuelles Objekt gibt, wird das aktuelle Objekt gelöscht und das Objekt hinter dem gelöschten Objekt wird zum aktuellen Objekt. Wird das Objekt, das am Ende der Liste steht, gelöscht, gibt es kein aktuelles Objekt mehr. Wenn die Liste leer ist oder es kein aktuelles Objekt gibt, bleibt die Liste unverändert.