18. Informatik – Fachbezogene Hinweise und Thematische Schwerpunkte für die schriftliche Abiturprüfung 2027

A. Fachbezogene Hinweise

Grundlage für die schriftliche Abiturprüfung 2027 in Niedersachsen sind die Einheitlichen Prüfungsanforderungen in der Abiturprüfung Informatik (EPA vom 01.12.1989 i.d.F. vom 05.02.2004), konkretisiert durch das Kerncurriculum Informatik für das Gymnasium – gymnasiale Oberstufe, die Gesamtschule – gymnasiale Oberstufe und das Kolleg (KC, 2017).

Entsprechend der Ausführungen des Kerncurriculums zu einheitlichen Darstellungsformen und Funktionsumfängen (KC, 2017, Abschnitt 3.5) sind die durch die „Ergänzenden Hinweise zum Kerncurriculum Informatik für die gymnasiale Oberstufe am Gymnasium und an der Gesamtschule sowie für das Kolleg“ mit Stand vom Juni 2025 vorgenommenen Festlegungen für die schriftliche Abiturprüfung 2027 verbindlich zu berücksichtigen.

B. Hinweise zu den Prüfungsaufgaben

Sämtliche im Kerncurriculum (KC, 2017) genannten inhaltlichen und prozessorientierten Kompetenzen sind für die schriftliche Abiturprüfung verbindlich. Dies gilt insbesondere auch für die Kompetenzen, die in den Lernfeldern für die Einführungsphase ausgewiesen sind.

Für das erhöhte und für das grundlegende Anforderungsniveau gilt: Jede **Prüfungsaufgabe** besteht aus drei Aufgaben. Den Prüflingen werden jeweils Aufgaben aus zwei Blöcken (1 und 2) zur Auswahl vorgelegt. Aus Block 1 ist genau eine Aufgabe zur Bearbeitung auszuwählen. Aus Block 2 sind genau zwei der drei Aufgaben auszuwählen. Andere Kombinationen sind nicht zulässig.
Jede Aufgabe aus Block 1 umfasst 50% der insgesamt zu erreichenden Bewertungseinheiten (BE). Jede der drei Aufgaben im Block 2 umfasst 25% der insgesamt zu erreichenden BE.

Block 1 (50% der BE)
Aufgabe 1A
Aufgabe 1B

Eine von zwei Aufgaben ist zu wählen.

Block 2 (50% der BE)
Aufgabe 2A
Aufgabe 2B
Aufgabe 2C

Zwei von drei Aufgaben sind zu wählen.

C. Sonstige Hinweise

- Die Aufgabenstellungen enthalten keinen Code in einer konkreten Programmiersprache.
- Aufgaben, die die Implementierung in einer konkreten Programmiersprache erfordern, sind von den Schülerinnen und Schülern in Java oder einer anderen objektorientierten Sprache mit imperativem Kern zu bearbeiten.
- Es werden keine Aufgaben gestellt, für die der Einsatz eines Rechners erforderlich ist.

Hilfsmittel

- Die in der Anlage zu diesen Hinweisen dokumentierten Informationen sind in ausgedruckter Form als Hilfsmittel zulässig.
- Die Verwendung eines Taschenrechners oder einer Formelsammlung ist in der schriftlichen Abiturprüfung im Fach Informatik nicht zulässig.

Anlage:

Hilfsmittel für Prüflinge zur Verwendung in der schriftlichen Abiturprüfung im Fach Informatik

Umgang mit Zeichen und Zeichenketten

Für die Arbeit mit Zeichenketten ist der Umfang der zu verwendenden Operationen auf die folgenden eingeschränkt. Die Verwendung von darüber hinausgehenden Zeichenkettenoperationen ist im Rahmen von zentralen Prüfungsaufgaben nicht zulässig.

- Bestimmen der Länge einer Zeichenkette
- Auslesen eines Zeichens an einer bestimmten Position
- Auslesen einer Teilzeichenkette in einem bestimmten Bereich
- Verbinden von zwei Zeichenketten zu einer
- Prüfen des Inhalts von zwei Zeichenketten auf Gleichheit
- Prüfen einer Zeichenkette auf eine Teilzeichenkette
- Lexikographisches Vergleichen von zwei Zeichenketten
- Bestimmen des ASCII-Werts (Dezimalzahl) zu einem Zeichen (und umgekehrt)

Umgang mit mathematischen Operationen

Für die Arbeit mit mathematischen Operationen zur Division und Modulo-Berechnung gelten für programmiersprachenunabhängige Betrachtungen die folgenden Vereinbarungen:

- Der Operator `/` steht bei der Verwendung mit zwei ganzen Zahlen für die ganzzahlige Division ohne Rest, wenn dies in der Aufgabenstellung nicht anders festgelegt ist.
- Der Operator `mod` steht für den (kleinsten, positiven) Rest, der bei der ganzzahligen Division von zwei ganzen Zahlen auftritt.

Bei Aufgabenstellungen zur Implementierung ist von den Prüflingen bei der Verwendung mathematischer Operationen die korrekte Syntax der verwendeten Programmiersprache zu nutzen.

Die Prüflinge müssen die Erzeugung von ganzzahligen Zufallszahlen in einem vorgegebenen Bereich und das ganzzahlige Runden in der im Unterricht verwendeten Programmiersprache beherrschen.

Umgang mit statischen Reihungen

In programmiersprachenunabhängigen Betrachtungen beginnt die Nummerierung der Elemente einer statischen Reihung mit dem Index 0.

Bei zweidimensionalen statischen Reihungen erfolgt die Notation der Indizes im Normalfall in der Reihenfolge [Zeile][Spalte], wenn die zugehörigen Daten in einer entsprechenden Tabelle vorliegen. In bestimmten Sachzusammenhängen ist auch eine andere Notation möglich. Dies ist in zentralen Prüfungsaufgaben dann deutlich beschrieben.

Statische Reihungen sind iterierbar, so dass die Verwendung einer for-each-Schleife zum elementweisen Auslesen einer statischen Reihung möglich ist.

Operationen für dynamische Reihungen, Stapel, Schlangen und Binärbäume

Dynamische Reihung

Die Nummerierung der Elemente der dynamischen Reihung beginnt mit dem Index 0. Dynamische Reihungen sind iterierbar, so dass die Verwendung einer for-each-Schleife zum elementweisen Auslesen einer dynamischen Reihung möglich ist.

`DynArray()`
Eine leere dynamische Reihung wird angelegt.

`isEmpty(): Wahrheitswert`
Wenn die dynamische Reihung kein Element enthält, wird der Wert `wahr` zurückgegeben, sonst der Wert `falsch`.

`getItem(index: Ganzzahl): Inhaltstyp`
Der Inhalt des Elements an der Position `index` wird zurückgegeben.

`append(inhalt: Inhaltstyp)`
Ein neues Element mit dem übergebenen Inhalt wird am Ende der dynamischen Reihung angefügt.

`insertAt(index: Ganzzahl, inhalt: Inhaltstyp)`
Ein neues Element mit dem übergebenen Inhalt wird an der Position `index` in die dynamische Reihung eingefügt. Das Element, das sich vorher an dieser Position befunden hat, und alle nachfolgenden werden nach hinten verschoben. Entspricht der Wert von `index` der Länge der dynamischen Reihung, so wird ein neues Element am Ende der dynamischen Reihung angefügt.

`setItem(index: Ganzzahl, inhalt: Inhaltstyp)`
Der Inhalt des Elementes an der Position `index` wird durch den übergebenen Inhalt ersetzt.

`delete(index: Ganzzahl)`
Das Element an der Position `index` wird entfernt. Alle folgenden Elemente werden um eine Position nach vorne geschoben.

`getLength(): Ganzzahl`
Die Anzahl der Elemente der dynamischen Reihung wird zurückgegeben.

Stapel

`Stack()`
Ein leerer Stapel wird angelegt.

`isEmpty(): Wahrheitswert`
Wenn der Stapel kein Element enthält, wird der Wert `wahr` zurückgegeben, sonst der Wert `falsch`.

`top(): Inhaltstyp`
Der Inhalt des obersten Elements des Stapels wird zurückgegeben, das Element aber nicht entnommen.

`push(inhalt: Inhaltstyp)`
Ein neues Element mit dem übergebenen Inhalt wird oben auf den Stapel gelegt.

`pop(): Inhaltstyp`
Das oberste Element des Stapels wird entnommen. Der Inhalt dieses Elements wird zurückgegeben.

Schlange

`Queue()`
Eine leere Schlange wird angelegt.

`isEmpty(): Wahrheitswert`
Wenn die Schlange kein Element enthält, wird der Wert `wahr` zurückgegeben, sonst der Wert `falsch`.

`head(): Inhaltstyp`
Der Inhalt des vordersten Elements der Schlange wird zurückgegeben, das Element aber nicht entnommen.

`enqueue(inhalt: Inhaltstyp)`
Ein neues Element mit dem angegebenen Inhalt wird am Ende an die Schlange angehängt.

`dequeue(): Inhaltstyp`
Das vorderste Element der Schlange wird entnommen. Der Inhalt dieses Elements wird zurückgegeben.

Binärbaum

`BinTree()`
Ein leerer Baum wird erzeugt. Er besitzt keinen Inhalt und keine Teilbäume.

`BinTree(inhalt: Inhaltstyp)`
Ein Baum wird erzeugt. Die Wurzel erhält den übergebenen Inhalt als Wert. Der Baum besitzt jeweils einen leeren Baum als linken und rechten Teilbaum.

`isEmpty(): Wahrheitswert`
Wenn der Baum ein leerer Baum ist, wird der Wert `wahr` zurückgegeben, sonst der Wert `falsch`.

`getItem(): Inhaltstyp`
Die Operation gibt den Inhaltswert der Wurzel des Baumes zurück.

`setItem(inhalt: Inhaltstyp)`
Die Wurzel des Baums erhält den übergebenen Inhalt als Wert.
Bei einem leeren Baum wird zusätzlich als linker und rechter Teilbaum jeweils ein leerer Baum gesetzt.

`isLeaf(): Wahrheitswert`
Wenn der Baum jeweils einen leeren Baum als linken und rechten Teilbaum besitzt, also ein Blatt ist, wird der Wert `wahr` zurückgegeben, sonst der Wert `falsch`.

`getLeft(): Binärbaum`
Die Operation gibt den linken Teilbaum zurück.

`setLeft(b: Binärbaum)`
Der übergebene Baum wird als linker Teilbaum gesetzt.

`getRight(): Binärbaum`
Die Operation gibt den rechten Teilbaum zurück.

`setRight(b: Binärbaum)`
Der übergebene Baum wird als rechter Teilbaum gesetzt.

`setEmpty()`
Der Baum wird zu einem leeren Baum, d. h. er besitzt keinen Inhalt und keine Teilbäume.

Datenbankabfragen

SELECT-Anweisung:

```
SELECT [DISTINCT] * | spalte1 [AS s1], spalte2 [AS s2], …
FROM tabelle1 [t1], tabelle2 [t2], …
[WHERE bedingung1 (AND | OR) bedingung2 (AND | OR) … ]
[GROUP BY spalte1, spalte2, …
[HAVING gruppenBedingung1 (AND | OR) gruppenBedingung2 (AND | OR) …]]
[ORDER BY spalte1 [ASC | DESC], spalte2 [ASC | DESC], …]
[LIMIT anzahl]
```

Angaben in eckigen Klammern sind optional. Spalten können Attribute, Aggregatfunktionen oder Berechnungen sein. Bei `GROUP BY` und `ORDER BY` ist auch die Angabe eines Alias möglich.

Operatoren für Berechnungen: `+`, `-`, `*`, `/`

Operatoren für Vergleiche in Bedingungen: `=`, `!=` (ungleich), `>`, `<`, `>=`, `<=`, `NOT`, `IS NULL`, `IN`, `BETWEEN`, `LIKE` (mit den Platzhaltern `_` (für genau ein Zeichen) und `%` (für beliebig viele Zeichen))

Aggregatfunktionen: `AVG`, `COUNT`, `MAX`, `MIN`, `SUM`

Hinweis zur Verwendung von `GROUP BY`: Bei einer Gruppierung müssen alle Spalten in der SELECT-Klausel entweder von einer Aggregatfunktion umschlossen oder explizit in der GROUP BY-Klausel aufgezählt sein.

Notation von Bitfolgen nach dem (7,4)-Hamming-Code

Die Daten- und Prüfbits im (7,4)-Hamming-Code werden in der Reihenfolge `p0 p1 d0 p2 d1 d2 d3` notiert.

Die Kontrollgruppen sind entsprechend der folgenden Abbildung zusammengesetzt:

	Datenbits			
Prüfbit	d0	d1	d2	d3
p0	x	x	-	x
p1	x	-	x	x
p2	-	x	x	x

Die Festlegung der Prüfbits erfolgt auf Basis einer geraden Parität.

ASCII-Tabelle / Vigenère-Verschlüsselungs-Quadrat / Alphabet mit Nummerierung

Auszug aus der ASCII-Tabelle

dez	char
…	…
32	Leerz.
33	!
34	"
35	#
36	$
37	%
38	&
39	'
40	(
41	)
42	*
43	+
44	,
45	-
46	.
47	/

dez	char
48	0
49	1
50	2
51	3
52	4
53	5
54	6
55	7
56	8
57	9
58	:
59	;
60	<
61	=
62	>
63	?
64	@

dez	char
65	A
66	B
67	C
68	D
69	E
70	F
71	G
72	H
73	I
74	J
75	K
76	L
77	M
78	N
79	O
80	P
81	Q

dez	char
82	R
83	S
84	T
85	U
86	V
87	W
88	X
89	Y
90	Z
91	[
92	\
93	]
94	^
95	_
96	`
97	a
98	b

dez	char
99	c
100	d
101	e
102	f
103	g
104	h
105	i
106	j
107	k
108	l
109	m
110	n
111	o
112	p
113	q
114	r
115	s

dez	char
116	t
117	u
118	v
119	w
120	x
121	y
122	z
123	{
124	|
125	}
126	~
…	…

Vigenère-Verschlüsselungs-Quadrat

Schlüsselbuchstabe \ Klartextbuchstabe	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Alphabet mit Nummerierung

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**
A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**