gds2 Gottlieb-Daimler-Schule 2
Technisches Schulzentrum Sindelfingen
mit Abteilung Akademie für Datenverarbeitung

1 Einfache Konsolprogramme; Variablendeklaration und Initialisierung; Ein- und Ausgabe

Ziel: sauber strukturierter Programmcode, Umgang mit Variablen, Interpretation der Compilermeldungen, Rechenoperationen, Umsetzen der erstellten Struktogramme in Java-Programme

1.1. Erstellen Sie mit einem ASCII-Editor (z.B. Notepad++) eine Quelldatei erstesProgramm.java und speichern Sie diese in Ihrem privaten Ordner für Quelldateien ab.

Erfassen Sie in dieser Datei den Quelltext eines Programms, das den Text "Hurra, jetzt bin ich ein JAVA-Programmierer" am Bildschirm ausgibt.

Übersetzen und starten Sie das Programm in einer Windows „Eingabeaufforderung“. (Vorgehensweise s. Informationsblatt „Bestandteile eines Java Programms“ S. 2)

1.2. Erstellen Sie ein Java-Programm nach folgenden Vorgaben:

- deklarieren Sie zu jedem einfachen Datentyp den Sie kennen jeweils eine Variable
- weisen Sie den Variablen gültige Werte zu. Verwenden Sie dabei die unterschiedlichen Möglichkeiten der Variableninitialisierung
- geben Sie die Variablen am Bildschirm aus
- weisen Sie den Variablen auch ungültige Werte zu. Was passiert beim Compilieren? Achten Sie auf die Fehlermeldungen.

1.3. Erstellen Sie ein Java-Programm nach folgenden Vorgaben:

- deklarieren Sie zwei Variablen `iZahl1` und `iZahl2` vom Datentyp int (Ganzzahl)
- weisen Sie den Variablen beliebige gültige Werte zu.
- deklarieren Sie eine Variable mit dem Namen `iErgebnis` vom Datentyp int
- mit `iZahl1` und `iZahl1` sollen nacheinander die vier Grundrechenarten und die Modulo-Operation ausgeführt, das Ergebnis in der Variablen `iErgebnis` gespeichert und direkt am Bildschirm ausgegeben werden.

Erweiterung der Aufgabe:

- deklarieren Sie drei Variablen `dErgebnis, dZahl1` und `dZahl2` vom Datentyp double (Gleitpunktzahl)
- führen Sie die gleichen Rechenoperationen wie mit den int Variablen durch und schauen Sie, ob sich die Ergebnisse verändern?

1.4. Implementieren Sie das Struktogramm von Aufgabe 1.5 von Lsg_Struktogramm.pdf. (Implementieren bedeutet programmieren in einer Programmiersprache, also in Java.)

1.5. Implementieren Sie das Struktogramm von Aufgabe 1.7 von Lsg_Struktogramm.pdf.

1.6. Flächenberechnung

Geben Sie das folgende Java-Programm ein.
Überlegen Sie sich, wie die Ausgabe des Programmes aussehen könnte!

```
package strProgAufg;

import input.Eingabe;

public class A0106Flaechenberechnung
{
	public static void main(String[] args)
	{
		/* Variablendeklaration */
		double seiteA, seiteB, flaeche;

		/* Eingabe */
		seiteA = Eingabe.getDouble("Seitenlänge a = ");
		seiteB = Eingabe.getDouble("Seitenlänge b = ");

		/* Berechnung */
		flaeche = seiteA * seiteB;

		/* Ausgabe */
		System.out.println("Fläche: " + flaeche + " m²");
	}
}
```

Führen Sie das Programm aus!

Welche Aufgabe hat die Methode `Eingabe.getDouble`?

Welche Aufgabe hat das + Zeichen bei der Ausgabe in der Methode `System.out.println`?

1.7. Erstellen Sie ein Programm zur Berechnung des Wasserinhalts eines Schwimmbeckens.

Außerdem soll der Preis für eine Schwimmbeckenfüllung berechnet und ausgegeben werden!
Der Benutzer muss dazu folgende Daten des Schwimmbeckens eingeben:

- Länge (l), Breite (b) und Wasserhöhe (h) in Meter.
- Der Wasserpreis (preis) soll 4,33 €/m^3 betragen.

a) Das Volumen des Schwimmbeckens (***V = l · b · h***) und der Preis für eine Schwimmbeckenfüllung sollen auf dem Bildschirm ausgegeben werden!
b) Berechnen Sie, welchen Durchmesser (***d = 2· r***) ein kreisrundes Schwimmbecken haben müsste (***V = π · r²· h***), damit in dieses die gleiche Wassermenge wie beim rechteckigen Schwimmbad hineinpassen würde!
Hinweis: Die Wurzel berechnet man in Java durch: ***wurzel = Math.sqrt(wert);***
c) Formatieren Sie alle entsprechenden Ausgaben so, dass auf 2 Nachkommastellen gerundet wird.

2 Rechnen in Java

Ziel: Benutzung geeigneter Datentypen, korrekte Typkonvertierung, Verständnis für Rechenvorgänge mit den Datentypen

2.1. Implementieren Sie das Struktogramm von Aufgabe 1.8 von Lsg_Struktogramm.pdf. Formatieren Sie die Ausgabe so, dass die Ausgabe auf eine Nachkommastelle gerundet wird.

2.2. Ein kleines Programm soll den Durchschnittsverbrauch eines Autos berechnen. Der Benutzer kann die gefahrene Strecke in km und den dabei verbrauchten Treibstoff in Liter eingeben. Das Programm gibt dann den Durchschnittsverbrauch Liter/(100 km) aus. Formatieren Sie die Ausgabe so, dass die Ausgabe auf eine Nachkommastelle gerundet wird.

2.3. Auf der 7. Etappe der Tour de France 2011 benötigte der Sieger 5:38:53 Std. Wie viele Stunden war er unterwegs? Das Programm soll die Dauer in Stunden als eine Dezimalzahl berechnen und am Bildschirm ausgeben.
Diese Etappe war 218,5 km lang. Wie hoch war die Durchschnittsgeschwindigkeit des Siegers in km/h? Das Programm soll sie berechnen und auf 2 Nachkommastellen genau am Bildschirm ausgeben.
Die Gesamtlänge der Tour betrug 3430 km mit insgesamt 21 Etappen.
- Wie lang war eine Etappe im Schnitt?
- Wie viel Prozent der Gesamtstrecke war das?
- Es wurden 22 Teams eingeladen, jedes Team besteht aus 9 Fahrern. 32 Fahrer schieden vorzeitig aus. Wieviel Prozent der Fahrer erreichten das Ziel in Paris?

2.4. Schreiben Sie ein Programm, das vom Benutzer die Eingabe eines Großbuchstabens anfordert. Das Programm gibt auf dem Bildschirm aus, der wievielte Buchstabe im Alphabet der eingegebene ist.
Hinweis: In Java kann mit Char- Werten gerechnet werden, d.h. über die Differenz kann der Abstand zwischen zwei Buchstaben ermittelt werden.
Erweiterung: Das Programm soll in jedem Fall funktionieren, egal ob Groß- oder Kleinbuchstaben eingegeben werden. Hilfreich ist hierfür die Methode `Character.toUpperCase(char Zeichen)`, die als Parameter ein Zeichen erwartet und als Rückgabewert den entsprechenden Großbuchstaben liefert.
Bsp.: `char cGroß; cGroß = Character.toUpperCase('a') // cGroß ← 'A'`

2.5. Geben Sie die genaue Ausgabe auf dem Bildschirm an oder eine Begründung, warum keine Ausgabe erfolgen kann:

```
int    iZahl = 10  , iErgebnis = 0;
byte   bZahl = 80  , bErgebnis = 0;
long   lZahl = 34;
float  fZahl = 4.5f, fErgebnis = 0.0f;
double dZahl = 12.5;
short  sZahl = 30  , sErgebnis = 0;
```

		Ergebnis	
1	`iErgebnis = iZahl / 4;`		
2	`bErgebnis = lZahl - 10;`		
3	`fErgebnis = lZahl / iZahl * fZahl;`		
4	`fErgebnis = fZahl * lZahl / iZahl;`		
5	`fErgebnis = dZahl * lZahl / iZahl;`		
6	`fErgebnis = dZahl * (float)iZahl;`		
7	`iErgebnis = (int)(fZahl/iZahl);`		
8	`sErgebnis = bZahl*dZahl;`		
9	`fErgebnis = (fZahl /(int)dZahl)+10.5;`		
10	`bErgebnis = (byte)sZahl*80;`		
11	`iErgebnis = (int)((10/3.0)*10);`		
12	`fErgebnis = fZahl * (lZahl / iZahl);`		

2.6. Schreiben Sie ein Programm mit folgender Funktion: Der Benutzer kann eine Kommazahl eingeben, die eine Zeitspanne in Stunden darstellt. Das Programm gibt daraufhin diese Zeitspanne in der Darstellung hh:mm:ss,ms am Bildschirm aus.

Beispiel: Zeitspanne in h = 3.8284
Die Zeitspanne entspricht 3h 49‘ 42,240 s

3 Verzweigung

3.1. Implementieren Sie das Struktogramm von Aufgabe 2.1 von Lsg_Struktogramm.pdf. Achten Sie auf die Wahl eines geeigneten Datentyps.

3.2. Implementieren Sie das Struktogramm von Aufgabe 2.2 von Lsg_Struktogramm.pdf. Formatieren Sie die Ausgabe so, dass die Ausgabe auf zwei Nachkommastelle gerundet wird.

3.3. Um die Auswertung von Wettkampfergebnissen für ein Sportabzeichen im Weitsprung besser vergleichen zu können, sollen die Sprungergebnisse mit Hilfe eines Computerprogramms ausgewertet werden:

	Gold	**Silber**	**Bronze**
männlich	über 5,70 Meter	über 4,70 bis 5,70 Meter	über 3,50 bis 4,70 Meter
weiblich	über 5,10 Meter	über 4,20 bis 5,10 Meter	über 3,10 bis 4,20 Meter

- Der Benutzer soll zunächst eingeben, ob er männlich (m) oder weiblich (w) ist. Gibt der Benutzer hier etwas Falsches ein (d.h. etwas anderes als 'm', 'M', 'w' oder 'W'), dann erhält er eine Fehlermeldung.
- Dann gibt der Benutzer seine Sprungweite ein und das Programm zeigt an, welche Auszeichnung der Sportler erhält.

3.4. Es soll ein Programm erstellt werden, bei dem der Benutzer aufgefordert wird, ein Zeichen einzugeben. Anschließend soll ausgegeben werden, ob das Zeichen

- eine Dezimalziffer
- ein Buchstabe oder
- ein anderes Zeichen ist.

Der Quelltext kann kürzer werden, wenn man die vorhandene Methode `Character.toLowerCase(cZeichen)` verwendet. Sie bekommt als Parameter einen char-Wert und gibt einen char-Wert zurück. Ist `cZeichen` ein Großbuchstabe, so wird dieser in einen Kleinbuchstaben umgewandelt und zurückgegeben. Ansonsten wird einfach `cZeichen` wieder zurückgegeben.

3.5. Implementieren Sie das Struktogramm von Aufgabe 2.8 von Lsg_Struktogramm.pdf.

3.6. Erstellen Sie ein Programm, das eine quadratische Gleichung der Form:
$ax^2 + bx + c = 0$ löst.
Die Koeffizienten `a, b, c` werden vom Benutzer eingegeben. Das Programm berechnet die Diskriminante und zeigt diese an. Dann wird x_1 und x_2 bzw. eine Lösung `x` berechnet und ausgegeben oder eine der folgenden Meldungen angezeigt:
- „a darf nicht 0 sein!"
- „Die Gleichung hat keine reelle Lösung!"

Achtung: Entwickeln Sie die Lösung zunächst in einem Struktogramm und führen einen Schreibtischtest durch.
(Testwerte: a = 2, b = -8, c = 8; → 1 Lösung: x = 2
a = 2, b = 4, c = 1,5; → 2 Lösungen: x1 = -1,5; x2 = -0,5
a = 3, b = 2, c = 4; → keine reelle Lösung!

4 Schleifen

Wählen Sie jeweils den am besten geeigneten Schleifentyp

4.1. Implementieren Sie die Struktogramme von Aufgabe 4.1 a) und b) von Lsg_Struktogramm.pdf.

4.2. a) Ein Programm soll solange Temperaturmesswerte einlesen, bis der Wert 9999 als Endekennung eingegeben wird. Aus den Messwerten soll der Mittelwert berechnet werden, wobei generell nur die Messwerte berücksichtig werden sollen, die im Bereich zwischen - 40 und + 50 liegen. Alle anderen Werte werden als Fehlmessungen betrachtet.

b) Wie a) allerding soll jetzt aus den gültigen Messwerten der Maximalwert ermittelt werden.
Hinweis: Überlegen zu zunächst, wie Sie (ohne Computer) vorgehen würden, wenn Sie in einer viele Seiten langen Zahlenspalte den höchsten Wert ermitteln sollen.

4.3. Die Eulersche[1] Zahl e = 2,7182818284523… spielt in der Mathematik und vielen anderen Wissenschaften eine große Rolle. Diese Zahl kann mit Hilfe einer Reihensumme berechnet werden: **`e = 1 + 1/1 + 1/(1*2) + 1/(1*2*3) + ... + 1/(n!)`**

Möchte man in einem Programm die Zahl näherungsweise berechnen, sollte man sich vorher genau überlegen, wie die einzelnen Reihenglieder berechnet werden können. Am besten man schaut sich zunächst die ersten paar Schritte an und analysiert, welche Größen sich von Schritt zu Schritt verändern und wie Schritt 2 mit Schritt 1 und Schritt 3 mit Schritt 2 zusammenhängt. Das Ergebnis dieser Überlegung ist dann die allgemeine Angabe von Schritt i. Eine Formel, bei der ein Schritt auf den vorherigen in dieser Weise aufbaut, nennt man Rekursionsformel.

```
Startwerte:     Summe <- 1,0
                Summand <- 1,0/1

Schritt i=1:    Summe <- 1,0 + 1,0          (= 2,0)
                Summand <- 1,0/2            (= 0,5)

Schritt i=2:    Summe <- 2,0 + 0,5          (= 2,5)
                Summand <- 0,5/3            (= 0,1666...)

Schritt i=3:    Summe <- 2,5 + 0,1666       (= 2,666...)
                Summand <- 0,1666... /4     (= 0,041666...)


Allgemein:
Schritt i:    Summe <- Summe + Summand
              Summand <- Summand/(i+1)
```

Erstellen Sie ein Programm, das die Eulersche Zahl berechnet, wobei der Benutzer die Zahl der Schritte vorgibt.

[1] Sie wurde nach dem Schweizer Mathematiker Leonhard Euler benannt, der zahlreiche Eigenschaften von e beschrieb.

4.4. Erstellen Sie ein Programm, das die Kreiszahl π nach folgender Summenformel berechnet: **π = 4 * (1 – 1/3 + 1/5 – 1/7 + ...**

Gehen Sie für den Algorithmus-Entwurf wie in Aufgabe 4.3 vor.

Erstellen Sie das Programm, wobei die Anzahl der Schritte nicht fest vorgegeben ist. Die Schleife soll solange laufen, bis sich die Summe nur noch in der 10^{-9} Stelle verändert. Wie viele Schleifendurchläufe sind notwendig?

4.5. Implementieren Sie das Struktogramme von Aufgabe 4.14 von Lsg_Struktogramm.pdf. Benutzen Sie die formatierte Ausgabe, so dass eine Tabelle angezeigt wird:

```
Bitte geben Sie den Einzahlungsbetrag in Euro ein:
1000

          |    1.Jahr  |    2.Jahr  |    3.Jahr  |    4.Jahr  |    5.Jahr
----------+------------------------------------------------------------------
2% Zinsen |    1020,00 |    1040,40 |    1061,21 |    1082,43 |    1104,08
3% Zinsen |    1030,00 |    1060,90 |    1092,73 |    1125,51 |    1159,27
4% Zinsen |    1040,00 |    1081,60 |    1124,86 |    1169,86 |    1216,65
5% Zinsen |    1050,00 |    1102,50 |    1157,63 |    1215,51 |    1276,28
6% Zinsen |    1060,00 |    1123,60 |    1191,02 |    1262,48 |    1338,23
```

4.6. Erstellen Sie ein Programm, das eine Multiplikationstabelle wie nebenstehend am Bildschirm anzeigt

```
     |   1   2   3   4   5   6   7   8   9  10
-----+----------------------------------------
   1 |   1   2   3   4   5   6   7   8   9  10
   2 |   2   4   6   8  10  12  14  16  18  20
   3 |   3   6   9  12  15  18  21  24  27  30
   4 |   4   8  12  16  20  24  28  32  36  40
   5 |   5  10  15  20  25  30  35  40  45  50
   6 |   6  12  18  24  30  36  42  48  54  60
   7 |   7  14  21  28  35  42  49  56  63  70
   8 |   8  16  24  32  40  48  56  64  72  80
   9 |   9  18  27  36  45  54  63  72  81  90
  10 |  10  20  30  40  50  60  70  80  90 100
```

4.7. Weihnachten ist nicht mehr soweit entfernt.
a) Schreiben Sie ein Programm, das nach Eingabe der Anzahl der Äste (Trapeze) und der Anzahl der Zeilen pro Ast einen Weihnachtsbaum ausgibt (linkes Bild).

b) Der Baum soll nun noch mit Kerzen geschmückt werden (rechtes Bild).

```
     *
    ***
   *****
    ***
   *****
  *******
   *****
  *******
 *********
  *******
 *********
***********
 *********
***********
*************
```

```
     *
    ***
   i***i
    ***
   *****
  i*****i
   *****
  *******
 i*******i
  *******
 *********
i*********i
 *********
***********
i***********i
```

5 Methodenaufrufe

5.1. Implementieren Sie die Struktogramme von Aufgabe 4.5 und 3.6 von Lsg_Struktogramm.pdf in einer Klasse und testen Sie das Programm.

5.2. In der Java Dokumentation (https://docs.oracle.com/) finden Sie eine Beschreibung der Bibliotheksmethode Math.random(). Erkundigen Sie sich über die Funktionalität dieser Methode. Erstellen Sie eine Methode mit Namen `zZahl`, die folgenden Anforderungen genügt:
- Die Methode besitzt zwei ganzzahlige Parameter: iStartwert, iEndwert
- Die Methode liefert als Rückgabewert eine Zufallszahl z: iStartwert <= z <= iEndwert

Testen Sie Ihre Methode zZahl, indem Sie in der Main-Methode das Struktogramm von Aufgabe 4.9 von Lsg_Struktogramm.pdf implementieren.

5.3. Implementieren Sie das Struktogramm von Aufgabe 4.7 von Lsg_Struktogramm.pdf und verwenden Sie Ihre bereits erstellte Methode zZahl von Aufgabe 5.2

5.4. **Programm Bruchrechnen:** Erstellt werden soll ein Programm, das zwei Brüche addieren, subtrahieren, multiplizieren und dividieren kann

a) Im Hauptprogramm (main) sollen der Zähler (iZ1) und Nenner (iN1) des ersten Bruchs, sowie der Zähler (iZ2) und Nenner (iN2) des zweiten Bruchs als ganze Zahlen eingegeben werden.
Anschließend soll der Benutzer die Möglichkiet erhalten auszuwählen, ob er die Brüche addieren, subtrahieren, multiplizieren oder dividieren möchte.

b) Erstellen Sie eine Klasse Bruchrechnen, die die Methoden `addieren(), subtrahieren(), multiplizieren() und dividieren()` enthält und definieren Sie diese Methoden. Alle Methoden sollen als Übergabe-Parameter die Zähler und Nenner beider Brüche erhalten und keinen Rückgabewert zurückgeben. In den Methoden soll der Zähler und Nenner des Ergebnisses berechnet und auf dem Bildschirm ausgegeben werden!
Rufen Sie die Methoden an entsprechender Stelle im Hauptprogramm auf.

c) Ergänzen Sie die Klasse Bruchrechnen um die Methode `berechneGGT` (siehe Aufgabe 4.8 von Lsg_Struktogramm.pdf). Nutzen Sie diese Methode in den Methoden `addieren(), subtrahieren(), multiplizieren() und dividieren()` um den Ergebnisbruch vor der Ausgabe zu kürzen.

d) **Zusatzaufgabe**: Wenn eine Methode frei von Ein- und Ausgabe Anweisungen ist, dann kann sie flexibler angewendet werden. Deshalb soll das Programm so geändert werden, dass die Rechenmethoden das Ergebnis nicht mehr am Bildschirm anzeigen, sondern als String an den Aufrufer (hier main) zurückgeben. Zu diesem Zweck wird die Klasse Bruchrechnen um eine weitere Methode namens `ToString` ergänzt, die dann in den Rechenmethoden zur Erzeugung des Rückgabestring benutzt werden kann.
Die Methode besitzt folgenden Kopf (Signatur):

```
public static String ToString(int zaehler, int nenner)
```

Die Methode hat folgende Funktionalität:
- Bsp.: Ist der zaehler = 1 und der nenner = 2 dann ist die Rückgabe "1/2"
- Bsp.: Ist der zaehler = 3 und der nenner = 2 dann ist die Rückgabe "1 1/2"
- Bsp.: Ist der zaehler = 6 und der nenner = 2 dann ist die Rückgabe "3"

5.5. Erstellen Sie ein Programm, dass die Unicode-Tabelle im Bereich der darstellbaren Zeichen ab 0x20 bis 0xFF am Bildschirm wie folgt anzeigt:

```
Start des Programms Unicode Tabelle

   |  0|  1|  2|  3|  4|  5|  6|  7|  8|  9|  A|  B|  C|  D|  E|  F|
---------------------------------------------------------------------
 20|   |  !|  "|  #|  $|  %|  &|  '|  (|  )|  *|  +|  ,|  -|  .|  /|
 30|  0|  1|  2|  3|  4|  5|  6|  7|  8|  9|  :|  ;|  <|  =|  >|  ?|
 40|  @|  A|  B|  C|  D|  E|  F|  G|  H|  I|  J|  K|  L|  M|  N|  O|
 50|  P|  Q|  R|  S|  T|  U|  V|  W|  X|  Y|  Z|  [|  \|  ]|  ^|  _|
 60|  `|  a|  b|  c|  d|  e|  f|  g|  h|  i|  j|  k|  l|  m|  n|  o|
 70|  p|  q|  r|  s|  t|  u|  v|  w|  x|  y|  z|  {|  ||  }|  ~|  ▯|
 80|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|
 90|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|
 A0|   |  ¡|  ¢|  £|  ¤|  ¥|  ¦|  §|  ¨|  ©|  ª|  «|  ¬|  -|  ®|  ¯|
 B0|  °|  ±|  ²|  ³|  ´|  µ|  ¶|  ·|  ¸|  ¹|  º|  »|  ¼|  ½|  ¾|  ¿|
 C0|  À|  Á|  Â|  Ã|  Ä|  Å|  Æ|  Ç|  È|  É|  Ê|  Ë|  Ì|  Í|  Î|  Ï|
 D0|  Ð|  Ñ|  Ò|  Ó|  Ô|  Õ|  Ö|  ×|  Ø|  Ù|  Ú|  Û|  Ü|  Ý|  Þ|  ß|
 E0|  à|  á|  â|  ã|  ä|  å|  æ|  ç|  è|  é|  ê|  ë|  ì|  í|  î|  ï|
 F0|  ð|  ñ|  ò|  ó|  ô|  õ|  ö|  ÷|  ø|  ù|  ú|  û|  ü|  ý|  þ|  ÿ|
```

Zusatzaufgabe: Ändern Sie das Programm so ab, dass man durch Anpassen jeweils einer symbolischen Konstanten (final) die Spaltenzahl und die Spaltenbreite einfach verändern kann. Beispiel: Anzahl Spalten auf 8 und Spaltenbreite auf 6 geändert.

```
Start des Programms Unicode Tabelle

     |    0|    1|    2|    3|    4|    5|    6|    7|
-----------------------------------------------------------
  20|     |    !|    "|    #|    $|    %|    &|    '|
  28|    (|    )|    *|    +|    ,|    -|    .|    /|
  30|    0|    1|    2|    3|    4|    5|    6|    7|
  38|    8|    9|    :|    ;|    <|    =|    >|    ?|
  40|    @|    A|    B|    C|    D|    E|    F|    G|
  48|    H|    I|    J|    K|    L|    M|    N|    O|
  50|    P|    Q|    R|    S|    T|    U|    V|    W|
  58|    X|    Y|    Z|    [|    \|    ]|    ^|    _|
  60|    `|    a|    b|    c|    d|    e|    f|    g|
  68|    h|    i|    j|    k|    l|    m|    n|    o|
  70|    p|    q|    r|    s|    t|    u|    v|    w|
  78|    x|    y|    z|    {|    ||    }|    ~|    ▯|
  80|    ?|    ?|    ?|    ?|    ?|    ?|    ?|    ?|
  88|    ?|    ?|    ?|    ?|    ?|    ?|    ?|    ?|
  90|    ?|    ?|    ?|    ?|    ?|    ?|    ?|    ?|
  98|    ?|    ?|    ?|    ?|    ?|    ?|    ?|    ?|
  A0|     |    ¡|    ¢|    £|    ¤|    ¥|    ¦|    §|
  A8|    ¨|    ©|    ª|    «|    ¬|    -|    ®|    ¯|
  B0|    °|    ±|    ²|    ³|    ´|    µ|    ¶|    ·|
  B8|    ¸|    ¹|    º|    »|    ¼|    ½|    ¾|    ¿|
  C0|    À|    Á|    Â|    Ã|    Ä|    Å|    Æ|    Ç|
  C8|    È|    É|    Ê|    Ë|    Ì|    Í|    Î|    Ï|
  D0|    Ð|    Ñ|    Ò|    Ó|    Ô|    Õ|    Ö|    ×|
  D8|    Ø|    Ù|    Ú|    Û|    Ü|    Ý|    Þ|    ß|
  E0|    à|    á|    â|    ã|    ä|    å|    æ|    ç|
  E8|    è|    é|    ê|    ë|    ì|    í|    î|    ï|
  F0|    ð|    ñ|    ò|    ó|    ô|    õ|    ö|    ÷|
  F8|    ø|    ù|    ú|    û|    ü|    ý|    þ|    ÿ|
```