Universität Erlangen-Nürnberg
Lehrstuhl für Betriebssysteme

Diplomvorprüfung Oktober 1997
“Grundlagen der Informatik”
8. Oktober 1997

Arbeitszeit: 120 Minuten
Erreichbare Punktzahl: 80

Angaben zur Person:

Name: ....

Vorname: ....

Geburtsdatum: .... Matrikelnummer: ....

Nicht von der Kandidatin bzw. vom Kandidaten auszufüllen!

Bewertung:

Aufgabe	1	2	3	4	5	6	7	8	Σ
max.	6	4	10	20	4	6	10	20	80
Punktzahl									

Note: ____________________

Bemerkungen

- Jedes Klausurexemplar besteht aus 10 Seiten. Bitte prüfen Sie Ihr Exemplar auf Vollständigkeit und Druckqualität.
- Schreiben Sie die Lösungen zu den Aufgaben in die dafür freigehaltenen Zwischenräume auf den entsprechenden Seiten bzw. auf die unbedruckten Blattrückseiten. Sollte dieser Platz nicht ausreichen, können Sie sich von der Aufsicht zusätzliche **Leerblätter** in Ihr Klausurexemplar einheften lassen, **bevor Sie diese beschreiben**. Beschriebene Blätter und Schmierpapier können nicht eingeheftet werden und dürfen am Ende der Bearbeitungszeit nicht mit abgegeben werden.
- Versehen Sie alle zusätzlich eingehefteten Blätter mit Ihrem Namen und Ihrer Matrikelnummer.
- Schmierpapier liegt an Ihrem Platz bereit. Falls Sie zusätzliches Schmierpapier benötigen, bitte melden!
- Bevor Sie mit der Bearbeitung der Aufgaben beginnen:
  - Füllen Sie bitte die “Angaben zur Person” auf dem Deckblatt *gut leserlich* aus.
  - Legen Sie bitte Ihren Lichtbildausweis und Studentenausweis für die Kontrolle bereit.
  - Unterschreiben Sie bitte die folgende Erklärung.
- Als Hilfsmittel dürfen (außer Schreibmaterial) ein nicht programmierbarer Taschenrechner, sowie das Hilfsblatt “Assemblercodes” verwendet werden.
- Fragen zu den Prüfungsaufgaben können grundsätzlich **nicht** beantwortet werden.
- Sie dürfen den Raum nicht verlassen, **bevor** Ihre Personalien überprüft wurden.
- Bleiben Sie bitte am Ende der Bearbeitungszeit solange an Ihrem Platz sitzen, bis alle Klausurunterlagen eingesammelt sind.
- Falls Sie die Prüfung nach Erhalt der Prüfungsaufgaben aus gesundheitlichen Gründen abbrechen müssen, so muß Ihre Prüfungsunfähigkeit durch eine Untersuchung in der Universitätsklinik nachgewiesen werden.

Viel Erfolg!

Erklärung

Durch meine Unterschrift bestätige ich

- den Empfang der vollständigen Klausurunterlagen (10 Blätter)
- die Kenntnisnahme der obigen Informationen.

Ich bin damit einverstanden, daß mein Prüfungsergebnis unter Angabe der Matrikelnummer veröffentlicht wird:	**JA**	**NEIN**
	☐	☐

Erlangen, 8. Oktober 1997 ....
(Unterschrift)

Aufgabe 1 (6 Punkte)

Stellen Sie die folgenden Zahlen in dezimaler, hexadezimaler und oktaler Schreibweise dar (jeweils eine Schreibweise ist angegeben)! Der Rechenweg muß dargestellt werden!

1) 60890_{10} 2) 262544_8 3) $C89.8_{16}$

Aufgabe 2 (4 Punkte)

Berechnen Sie im Dualsystem (der Rechenweg muß dargestellt werden!):

$$(10110101_2 + 10011_2) * 10111_2$$

Aufgabe 3 (10 Punkte)

Geben Sie die Inhalte der Speicherzellen `fak`, `mak` und `bak` sowie der benötigten Register während und nach Ausführung des folgenden Assemblerprogramms an (in Dezimaldarstellung)!

```
0         LAD,  @fak
1         SUB,  #2
2         LAD,  (AC)
3         SPI,  @mak
4         LDX,  @bak
5   whl:  INK,
6         LIA,
7         SUB,  #3
8         JEZ,  @whl
9         JGZ,  @bye
10        SUB,  @mak
11        JLZ,  @lpx
12        JMP,  @whl
13  lpx:  LIA,
14        AND,  #6
15        LDI,
16        JMP,  @whl
17  bye:  HLT
18
19  fak:  DATA,  #21
20  mak:  DATA,  #0
21  bak:  DATA,  #1
```

Zeile	fak	mak	bak		

Aufgabe 4 (20 Punkte)

Übersetzen Sie das folgende C-Programm in die von Rembold angegebene hypothetische Assemblersprache! Verwenden Sie dabei für die Übergabe der Parameter an das Unterprogramm '`eual`' sowie für die Übergabe des Rückgabewertes an das Hauptprogramm den Kellerspeicher (Stack)! Desweiteren existiere ein Pseudo-Assembler-Befehl '`_input, @x`', welcher einen vom Benutzer über die Tastatur einzugebenden Wert in die Speicherzelle '`x`' lädt.

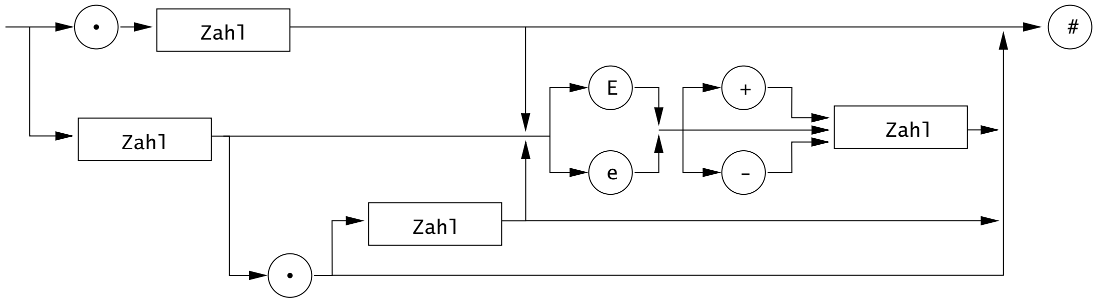
```
# include <stdio.h>
int eual(int a, int b)
{
   int c;
   while (a > 0)
   {
      if (a < b)
      {
         c = a;
         a = b;
         b = c;
      }
      a = a - b;
   }
   return b;
}

void main(void)
{
   int x, y, z;
   scanf("%d", &x);
   scanf("%d", &y);
   if (x>0 && y>0)
      z = eual(x,y);
}
```

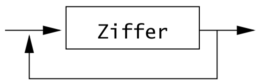
Aufgabe 5 (4 Punkte)

Gegeben seien die folgenden Syntaxdiagramme:

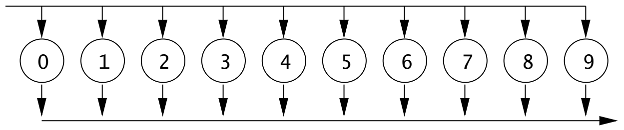
`Real-Konstante`



`Zahl`



`Ziffer`

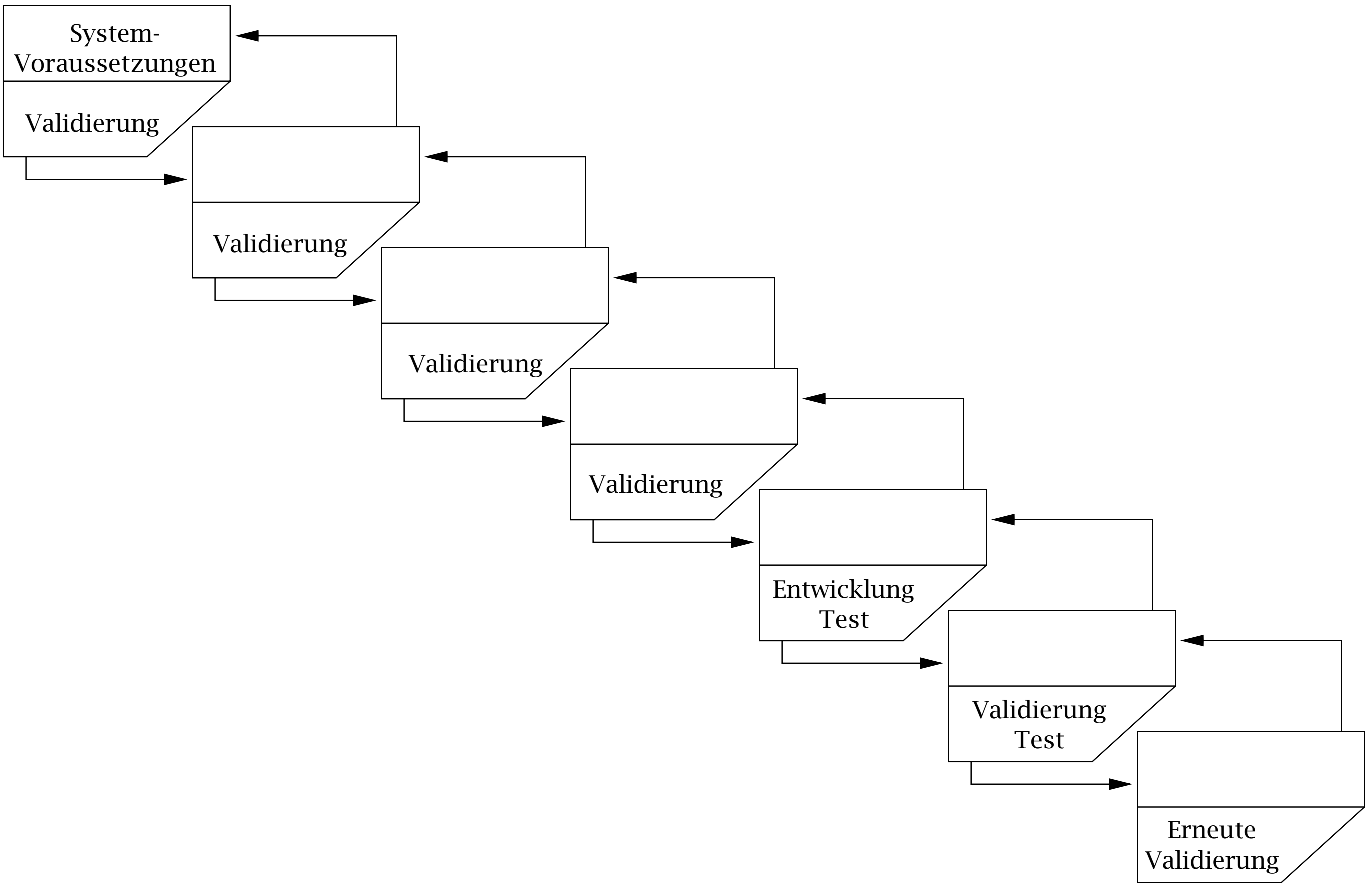


Welche der unten aufgeführten Zeichenfolgen sind demnach gültige Real-Konstanten? (Unterstreichen!) Falls die Zeichenfolge keine gültige Real-Konstante ist, markieren Sie das längste Teilstück von Beginn an, das noch ableitbar ist.

a) `.692E-73#` b) `397.34e4+20#` c) `-3.14115#` d) `59.22E-12#`

Aufgabe 6 (6 Punkte)

Tragen Sie in das unten skizzierte Gerüst des in der Vorlesung vorgestellten “Wasserfall”-Modells der Softwareentwicklung die fehlenden Vorgaben, durchzuführenden Tätigkeiten und Ergebnisse ein!



Aufgabe 7 (10 Punkte)

Im Arbeitsspeicher eines Rechensystems stehen zum Zeitpunkt $t = 0$ die Prozesse P_1, P_2 und P_3 in der unten gezeigten Anordnung.

Betriebs-system	P_1 5M	frei 5M	P_2 5M	frei 4M	P_3 3M	frei 6M
4	9	14	19	23	26	32

Zu den folgenden Zeitpunkten ergeben sich Änderungen:

Zeitpunkt	Aktion	Speicherbedarf
$t = 1$	neuer Prozeß: P_4	4M
$t = 2$	neuer Prozeß: P_5	6M
$t = 3$	neuer Prozeß: P_6	5M

Lassen sich diese Speicheranforderungen erfüllen, wenn

a) die "Best-Fit" Methode verwendet wird oder

b) die "First-Fit" Methode verwendet wird oder

c) die "Worst-Fit" Methode verwendet wird?

Zeichnen Sie in jedem Fall die resultierende Speicherbelegung (entweder am Schluß oder zu dem Zeitpunkt, an dem die erste Speicheranforderung nicht mehr befriedigt werden kann)! Verwenden Sie dazu die vorbereiteten Belegungspläne:

a)

Betriebs-system	P_1 5M		P_2 5M		P_3 3M	
4	9	14	19	23	26	32

b)

Betriebs-system	P_1 5M		P_2 5M		P_3 3M	
4	9	14	19	23	26	32

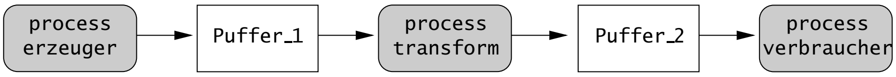
c)

Betriebs-system	P_1 5M		P_2 5M		P_3 3M	
4	9	14	19	23	26	32

Welche Maßnahme kann das Betriebssystem durchführen, falls eine Speicheranforderung im obigen Beispiel nicht erfüllt werden kann?

Aufgabe 8 (20 Punkte)

Betrachten Sie das folgende, schematisch dargestellte, erweiterte Erzeuger-Verbraucher Problem:



Der zusätzliche Prozeß `transform` liest Werte aus `Puffer_1` aus, verarbeitet sie anschließend weiter und schreibt die Resultate in `Puffer_2`, von wo diese dann vom Prozeß `verbraucher` ausgelesen werden. Die beiden Puffer besitzen jeweils eine endliche Kapazität von `max_elt` Einträgen.
Bei dem hier betrachteten Problem gilt es die folgenden Bedingungen zu erfüllen.

1. Der Erzeugerprozeß (`erzeuger`) darf nur dann einen Wert in `Puffer_1` schreiben, wenn die Kapazität dieses Puffers noch nicht erschöpft ist.
2. Der Prozess `transform` darf nur dann ein Element aus `Puffer_1` entnehmen, wenn dieser Puffer noch Elemente enthält.
3. `transform` darf solange Werte in `Puffer_2` eintragen, bis dieser seine Kapazitätsgrenze erreicht hat.
4. Der Prozeß `verbraucher` darf nur dann einen Wert aus `Puffer_2` auslesen, wenn dieser nicht leer ist.

Geben Sie eine Lösung dieses Problems mittels Semaphoren an! Verwenden Sie dabei für die Darstellung von Prozessen und Semaphoren die in der Vorlesung verwendete C-ähnliche Notation.