CT Übungsaufgaben
Modulare Programmierung und Linker

Aufgabe 1

Nennen Sie mindestens vier Punkte bei welchen die modulare Programmierung besonderen Nutzen bringt.

Thema	Nutzen
Teamarbeit	*Mehrere Entwickler können auf denselben Sourcen verteilt arbeiten*
Nützlich Aufteilung und Strukturierung von Programmen in konsistente Komponenten	*Ermöglicht Wiederverwendung von Komponenten*
Individuelle Verifikation einzelner Komponenten	*Zum Nutzen aller welche die Komponente verwenden*
Bibliotheken von mehrfach verwendbaren Funktionen und Typen	*Wiederverwendung anstelle von „das Rad neu erfinden“*

Weitere:

Mischen von Komponenten die in unterschiedlichen Programmiersprachen geschrieben wurden	*Komponenten können unabhängig von der Programmiersprache wiederverwendet werden*
Partielles Kompilieren	*Nur geänderte Komponenten müssen neu kompiliert werden.*

Aufgabe 2

Welche C Linkage haben die unterstrichenen Namen der gegebenen C Definitionen?

Code	External Linkage	Internal Linkage	No Linkage

`int square(int v) {`
`  return v * v;`
`}`
	x		

`int square(int v) {`
`  int res = v * v;`
`  return res;`
`}`
			x

`static int square(int v) {`
`  return v * v;`
`}`
		x	

Aufgabe 3

Ordnen Sie den Linker-Tasks 1 – 4 die entsprechenden Situationen a – d zu.

Linker-Task	Situation
1) Merge code sections	a) Modul A ruft eine Funktion aus Module B auf
2) Merge data sections	b) Modul A und Modul B enthalten Instruktionen
3) Resolve referenced symbols	c) Modul A und Modul B enthalten globale Daten
4) Relocate addresses	d) Verwendete Referenzen im Code müssen an die neue Lage der Symbole angepasst werden

1 – b
2 – c
3 – a
4 – d

Aufgabe 4

Beim Linken werden Object Files zu einem Executable File zusammengebunden. Damit dieses Binden erfolgreich ausgeführt werden kann, muss jedes Objekt File entsprechende Informationen enthalten.

Nennen Sie die Object File Sections welche die entsprechenden Informationen enthalten.

Object File Section Name	Enthaltene Information
Code Section	Instruktionen
Data Section	Globale Daten
Symbol Table	Liste der internen, importierten und exportierten Symbole
Relocation Table	Adressen der Instruktionen und Daten deren Zugriff auf Symbole im Laufe des Linkens angepasst werden müssen

Aufgabe 5

Vom C Programm zum Executable: Nennen und beschreiben Sie die vier Schritte (Tools), welche bei der Übersetzung eines C Programmes in ein ausführbares Executable File notwendig sind.

Name des Schrittes (Tools)	Was macht der Schritt?
Preprocessor	***Verarbeitet die Preprocessor Statements (z.B. #define, #include).*** ***Textprocessing: Ersetzen und Ergänzen von Inhalten.*** ***Ergebnis: Textfile mit modifiziertem C Sourcecode***
Compiler	***Übersetzt C Sourcecode in prozessorspezifische, symbolische Assemblerbefehle.*** ***Ergebnis: Textfile mit Assemblercode***
Assembler	***Übersetzt Assemblercode in binäre Maschinenbefehle.*** ***Ergebnis: binäres Objectfile mit Maschinenbefehlen, Daten, Symbol Tabelle, Relocation Tabelle.***
Linker	***Fügt mehrere Objectfiles zu einem ausführbaren Objektfile zusammen (Merge Sections), löst die Symbol Referenzen zwischen den einzelnen Objektfiles auf (Resolution) und passt Zugriffe auf Symbole an (Relocation).*** ***Ergebnis: Ausführbares Objektfile (Executable)***

Aufgabe 6

Folgender Ausschnitt einer Object File Symbol Table Section ist gegeben.

```
...
** Section #6 '.symtab' (SHT_SYMTAB)
 #  Symbol Name                Value      Bind  Sec  Type  Vis  Size
=========================================================================
 7  a                          0x00000000   Lc    4  Data  De   0x4
 8  b                          0x00000004   Lc    4  Data  De   0x4
11  main                       0x00000001   Gb    1  Code  Hi   0x14
12  square                     0x00000000   Gb  Ref  Code  Hi
...
```

Beantworten Sie folgende Fragen:

1) Welche Symbole stehen für Daten Adressen?

a, b

2) Welche Symbole stehen für Code Adressen?

main, square

3) Welche Symbole werden von diesem Modul importiert bzw. verwendet ohne dass sie in diesem Modul definiert sind?

square

4) Welche Symbole werden in diesem Modul definiert?

a, b, main

Aufgabe 7

Was ist eine Native-Compiler Tool Chain, was eine Cross-Compiler Tool Chain?

1) Native-Compiler Tool Chain:

Erzeugt Programme für die gleiche Architektur/Umgebung in welcher die Tool-Chain ausgeführt wird.

2) Cross-Compiler Tool Chain:

Erzeugt Programme für eine andere Architektur/Umgebung als jene in welcher die Tool-Chain ausgeführt wird.