

Vorname

Nachname

Matrikel-Nr

Aufgabe 1 (20 Punkte): Schreiben Sie (in C++) ein Unterprogramm namens **anzahl** entsprechend der folgenden Deklaration:

```

1 unsigned anzahlZiffern7(vector<unsigned> const & vu);
2 // Man stelle sich die Ganzzahlen im Vektor vu als Dezimalzahlen darge-
3 // stellt vor. Wie oft kommt in diesen Darstellungen die Ziffer 7 vor?
4 // Diese Funktion liefert die Antwort.
5 // Beispiel: Wenn ein Vektor v1 die vier Ganzzahlen 707, 123, 37, 2737
6 // enthaelt, dann ist anzahlZiffern7(v1) gleich 5. Falls v2 ein leerer
7 // Vektor ist, dann ist anzahlZiffern7(v2) natuerlich gleich 0.

```

Aufgabe 2 (20 Punkte): Schreiben Sie (in C++) ein Unterprogramm namens **plustere** entsprechend der folgenden Deklaration:

```

1 string plustere(string s);
2 // Liefert eine Kopie von s, in der zwischen je zwei Zeichen ein Blank
3 // ("Leerzeichen", space) eingefuegt wurde, aber nur dann, wenn
4 // mindestens eines der beiden Zeichen ein Buchstabe ist. Beispiele:
5
6 // s          | plustere(s)          | Laenge des Ergebnisses
7 // -----|-----|-----
8 // "abc"      | "a b c"             | 5 Zeichen
9 // "123"      | "123"               | 3 Zeichen
10 // "DM150"    | "D M 150"           | 7 Zeichen
11 // "14STDN17MIN" | "14 S T D N 17 M I N" | 19 Zeichen
12 // "X"        | "X"                 | 1 Zeichen
13 // ""         | ""                  | 0 Zeichen
14 // -----|-----|-----

```

Um festzustellen, ob ein Zeichen (vom Typ `char`) ein **Buchstabe** ist, koennen Sie die Funktion **isalpha** verwenden, z.B. so: `if (isalpha(s[i])) ...`. Die Funktion liefert **true**, wenn ihr Parameter ein Buchstabe ist und sonst liefert sie **false**.

Aufgabe 3 (20 Punkte): Betrachten Sie das folgende Unterprogramm:

```

1 void main() {
2     struct Verbund {
3         Verbund * z1;
4         Verbund * z2;
5         string s;
6         Verbund(Verbund * z1, Verbund * z2, string s) :
7             s(s), z1(z1), z2(z2) {}
8     }; // struct Verbund
9
10    Verbund * const P1 = new Verbund(NULL, NULL, "Alf");
11    P1->z2               = new Verbund(P1, NULL, "Bernd");
12    P1->z2->z2            = new Verbund(NULL, P1, "Carl");
13    P1->z1               = P1;
14    ...
15 } // main

```

Stellen Sie sich vor, dass der Ausfuehrer dieses Unterprogramm bis zur Zeile 14 ausgefuehrt hat. Wie sieht die Zeiger-Konstante **P1** und alles, "was an P1 dranhaengt", in diesem Moment aus? Stellen Sie die Zeiger-Konstante **P1** und alles, was an ihr dranhaengt, als **Bojen** dar.

Aufgabe 4 (20 Punkte): Schreiben Sie eine Funktions-Schablone namens `sum`, auf die folgende Beschreibung passt:

Schablonen-Parameter: Einer (ein Typ `T`)
Unterprogramm-Parameter: Drei (vom Typ `T`)
Rückgabe-Typ: Ebenfalls der Typ `T`.

Wenn man eine Instanz der Schablone `sum` aufruft, soll sie die **Summe** ihrer Parameter als Ergebnis liefern.

Damit man eine Instanz dieser Schablone `sum` wahlweise mit zwei oder drei (aktuellen) Parametern aufrufen kann, sollen Sie den dritten (formalen) Parameter mit einem geeigneten **Vorbesetzungswerten** versehen.

Aufgabe 5 (20 Punkte): Betrachten Sie das folgende C++-Programm namens `Prog01`:

```
1 // Datei Prog01.cpp
2 // -----
3 class K1 {
4 public:
5     float wert;
6 public:
7     K1(float w) {setWert(w);}
8     float getWert() {return wert;}
9     void setWert(float w) {wert = w;}
10 }; // class K1
11 // -----
12 class K2 {
13 public:
14     K1 * p1;
15     K1 * p2;
16 public:
17     K2(K1 * p1, K1 * p2) {
18         this->p1 = p1;
19         this->p2 = p2;
20     } // Konstruktor K2
21     float getWert1() {return p1->getWert();}
22     float getWert2() {return p2->getWert();}
23     void setWert1(float w) {p1->setWert(w);}
24     void setWert2(float w) {p2->setWert(w);}
25 }; // class K2
26 // -----
27 void main() {
28     K1 ob01(1.1F);
29     K2 ob02(&ob01, new K1(2.2F));
30     K2 ob03(&ob01, new K1(3.3F));
31     ...
32 } // main
33 // Ende der Datei Prog01.cpp
```

Stellen Sie sich den Moment vor, in dem der Ausführer damit beginnt, die Befehle in Zeile 31 auszuführen.

5.1. Wieviele Module existieren in diesem Moment?

5.2. Geben Sie die Namen dieser Module an. Falls ein Modul mehrere Namen hat, genügt es, wenn Sie **einen** dieser Namen angeben.

5.3. Geben Sie für jeden Modul `M` an, welche Elemente sich in `M` befinden. Geben Sie für jedes Element einen zusammengesetzten Namen an (der aus einem Modulnamen und dem Namen des betreffenden Elements besteht).

Lösung 1:

```

1 unsigned anzahlZiffern7(vector<unsigned> const & vu) {
2     // Man stelle sich die Ganzzahlen im Vektor vu als Dezimalzahlen darge-
3     // stellt vor. Wie oft kommt in diesen Darstellungen die Ziffer 7 vor?
4     // Diese Funktion liefert die Antwort.
5     // Beispiel: Wenn ein Vektor v1 die vier Ganzzahlen 707, 123, 37, 2737
6     // enthaelt, dann ist anzahlZiffern7(v1) gleich 5. Falls v2 ein leerer
7     // Vektor ist, dann ist anzahlZiffern7(v2) natuerlich gleich 0.
8
9     unsigned anzahl7 = 0;
10    for (int i=0; i<vu.size(); i++) {
11        unsigned n = vu[i];
12
13        while (n > 0) {
14            if (n % 10 == 7) anzahl7++;
15            n /= 10;
16        } // while
17    } // for
18    return anzahl7;
19 } // anzahlZiffern7

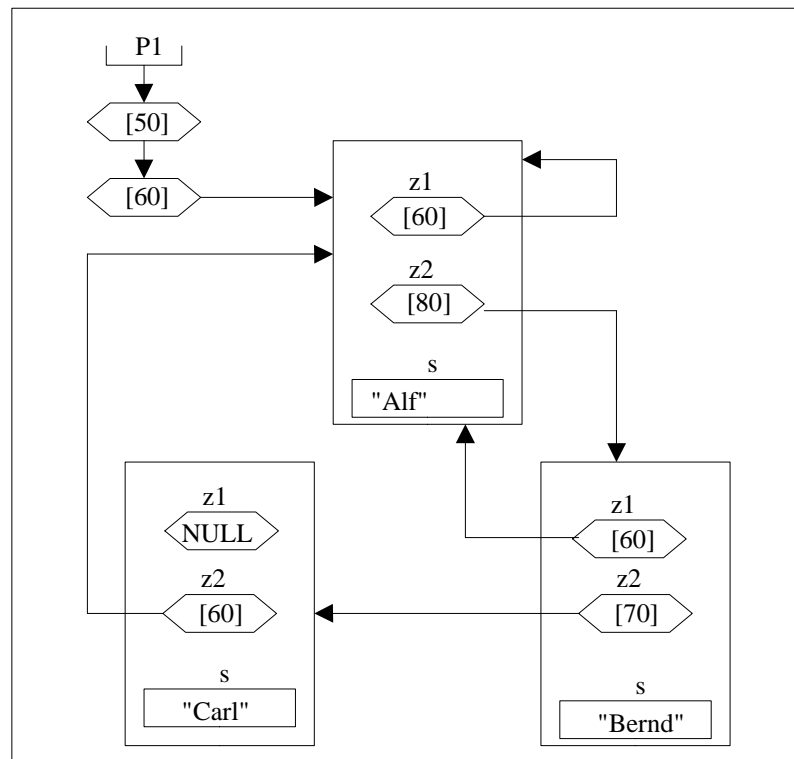
```

Lösung 2:

```

1 string plustere(string s) {
2     // Liefert eine Kopie von s, in der zwischen je zwei Zeichen ein Blank
3     // ("Leerzeichen", space) eingefuegt wurde, aber nur dann, wenn
4     // mindestens eines der beiden Zeichen ein Buchstabe ist. Beispiele:
5
6     // s          | plustere(s)          | Laenge des Ergebnisses
7     // -----|-----|-----
8     // "abc"      | "a b c"              | 5 Zeichen
9     // "123"      | "123"                | 3 Zeichen
10    // "DM150"     | "D M 150"            | 7 Zeichen
11    // "14STDN17MIN" | "14 S T D N 17 M I N" | 19 Zeichen
12    // "X"         | "X"                  | 1 Zeichen
13    // ""          | ""                   | 0 Zeichen
14    // -----|-----|-----
15
16    string erg;
17    char const BLANK = ' ';
18    for (int i=0; i+2<=s.size(); i++) { // Trick: i+2<=s.size() statt
19        // i <=s.size-2 (unsigned!)
20        erg += s[i];
21        if ( isalpha(s[i]) || isalpha(s[i+1]) ) {
22            erg += BLANK;
23        }
24    } // for
25    if (s.size() > 0) {
26        erg += s[s.size()-1];
27    }
28    return erg;
29 } // plustere

```

Lösung 3:**Lösung 4:**

```

1 template <typename T>
2 T sum(T a, T b, T c=0) {
3     return a + b + c;
4 } // sum

```

Lösung 5:

5.1., 5.2: Sieben Module: K1, K2, ob01, ob02, ob03, *ob02.p2, *ob03.p2

5.3.

Elemente des Moduls K1: K1::K1

Elemente des Moduls K2: K2::K2

Elemente des Moduls ob01:

ob01.wert, ob01.getWert, ob01.setWert

Elemente des Moduls ob02:

ob02.p1, ob02.p2, ob02.getWert1, ob02.getWert2, ob02.setWert1, ob02.setWert2

Elemente des Moduls ob03:

entsprechend wie ob02

Elemente des Moduls *ob02.p2:

ob02.p2->wert, ob02.p2->getWert, ob02.p2->setWert

Elemente des Moduls *ob03.p2:

entsprechend wie *ob02.p2