

2. Aufgabe: Grundlagen

Schreiben Sie ein Hauptprogramm, das die Kantenlänge eines Rechtecks von der Konsole (über cin) als reelle Zahl einliest und sowohl Umfang als auch Fläche als Ergebnis ausgibt (9P).

3. Aufgabe: Schleifen

a) Kreuzen Sie an, welche Schleifen es in C/C++ gibt (6P):
(Falsche Antworten geben 1 Punkt Abzug; minimale Punktzahl dieser Aufgabe sind 0 P)

1. for.....
2. repeat ... until.....
3. do ... while
4. if
5. foreach
6. when
7. else
8. while.....
9. forget.....
10. loop

b) Schreiben Sie nachfolgend eine Schleife, in der die Quadrate aller Zahlen von 100 bis 199 summiert werden und das Ergebnis nach der Schleife in einer unsigned int-Variablen steht. Verwenden Sie für die Bestimmung der Quadrate ein Makro SQR und definieren Sie dieses Makro vor Beginn des Programmfragments (10P):

4. Aufgabe: Array/Feld, Indizierung

Gegeben ist ein globales Feld mit den Monatstagen im Jahr:

```
unsigned int monatstage[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
```

a) Schreiben Sie eine Funktion ‚berechneTagImJahr‘, der Sie ein Datum mittels dreier Parameter ‚tag‘, ‚monat‘, ‚jahr‘ übergeben. Der Rückgabewert der Funktion soll die Anzahl der bis zu diesem Datum vergangenen Tage (inklusive des angegebenen Datums) bestimmen. Beachten Sie dabei, dass alle durch 4 teilbaren Jahre im Februar 29 Tage besitzen (Ausnahmen von dieser Regel sollen nicht berücksichtigt werden) (18P).

b) Skizzieren Sie eine Idee, wie man mit diesem Wert die Kalenderwoche bestimmen könnte (2P).

Beispiele:

berechneTagImJahr(20, 1, 2010) liefert den Wert 20.

berechneTagImJahr(10, 3, 2008) liefert den Wert 70.

5. Aufgabe: Ganzes Programm

Ergänzen Sie die Lücken im nachfolgenden Programmtext zu einem lauffähigen C++-Programm. Es hat die Aufgabe, zwei Zeichenketten, die mit dem Wert 0 enden, auf Gleichheit zu prüfen. Dabei soll die Groß- und Kleinschreibung keine Rolle spielen (Da jedem Buchstabensymbol ein Wert zugeordnet ist und die Buchstabensymbole sowohl der großen als auch der kleinen Buchstaben lückenlos und alphabetisch angeordnet sind, kann ein Kleinbuchstabe durch Subtraktion in einen Großbuchstaben geändert werden. Die Differenz ist für alle zugehörigen Buchstaben gleich und entspricht dabei der Differenz zweier zueinander gehörenden Buchstabensymbole, also z.B. 'a'-'A' == 'b'-'B'). (26P)

```
#include _____
```

```
using _____
```

```
// Umwandeln der gegebenen Zeichenkette in Großbuchstaben
```

```
void inGrossbuchstaben(char* acZeichenkette)
```

```
{
    _____; // Initialisierung der Laufvariablen iIndex
    char cAlphabetAbstand = 'a' - 'A';
    while(acZeichenkette[iIndex] != _____) // Zeichenkettenende erreicht?
    {
        if(acZeichenkette[iIndex] >= 'a' && _____ <= 'z') // Kleinbuchstabe?
        {
            acZeichenkette[iIndex] -= cAlphabetAbstand;
        }
        _____; // Erhöhung der Laufvariablen um 1
    }
}
```

```
// Prüfung der Zeichenketten auf Gleichheit
```

```
// Rückgabewert true: Zeichenketten sind gleich, Rückgabewert false: Zeichenketten sind ungleich.
```

```
bool istGleich(char* acZeichenkette1, char* acZeichenkette2)
```

```
{
    _____; // Initialisierung der Laufvariablen iIndex
    while(acZeichenkette1[iIndex] != 0 && acZeichenkette2[iIndex] != 0 &&
        acZeichenkette1[_____acZeichenkette2[_____]) // beide Zeichen gleich?
    {
        _____; // Erhöhung der Laufvariablen um 1
    }
    if(acZeichenkette1[iIndex] == 0 && acZeichenkette2[iIndex] == 0)
    { // Ende beider Zeichenkette erreicht!
        return _____;
    }
    return _____;
}
```

- Bitte beachten Sie auch die Rückseite -
- Lösen Sie die Aufgaben bitte auf dem Blatt -

```
int main()
{
    char acZeichenkette1[200];
    char acZeichenkette2[200];

    cout _____ "Zeichenkette1: ";
    cin _____ acZeichenkette1;
    inGrossbuchstaben(acZeichenkette1);

    cout _____ "Zeichenkette2: ";
    cin _____ acZeichenkette2;
    inGrossbuchstaben(acZeichenkette2);

    cout _____ "Zeichenketten sind ";

    if(_____ )
    {
        cout _____ "un";
    }
    cout _____ "gleich";

    return 0;
}
```

6. Aufgabe: Algorithmus

a) Was berechnet die nachfolgende Funktion bzw. was enthalten die Variablen iValue1 und iValue2 des Ergebnisses? Bitte beschreiben Sie die Funktionsweise möglichst abstrakt. (15 P)

b) Wann erhalten Sie ein nicht definiertes Ergebnis? (2P)

Hinweis: Testen Sie den Algorithmus anhand eines kleinen Arrays (z.B. der Größe 5) und beobachten Sie die Variablenwerte.

```
struct tResult
{
    int iValue1;
    int iValue2;
};

tResult unknown(int* aiInArray, unsigned int uiInAnzahl)
{
    tResult sResult;
    sResult.iValue1 = aiInArray[0];
    sResult.iValue2 = aiInArray[0];
    for(unsigned int uiIndex=0; uiIndex < uiInAnzahl; ++uiIndex)
    {
        if(sResult.iValue1 < aiInArray[uiIndex])
        {
            sResult.iValue1 = aiInArray[uiIndex];
        }
        sResult.iValue2 = sResult.iValue2 > aiInArray[uiIndex] ? aiInArray[uiIndex] : sResult.iValue2;
    }
    return sResult;
}
```