

Übung 5: Aufeinander folgende AUFRUFE

Heute ist zu verstehen, dass alle Funktionen fähig sind, Funktionen aufzurufen. Ansonsten sind Arrays wie üblich anwesend und Zeiger werden für verschiedene Aufgaben benutzt, nämlich: dynamische Speicherverwaltung und Parameterübergabe.



Verzeichnis von Begriffen

- | | |
|-------------------------|---------------------------|
| – Array | – Zeiger auf |
| – Parameter | – Funktionsaufruf |
| – Übergabe per Wert | – Dynamische Reservierung |
| – Übergabe per Referenz | – Dynamische Freigabe |

AUFGABE 0: Der Zweck des Programms ist den Betrag eines Vektors zu rechnen.

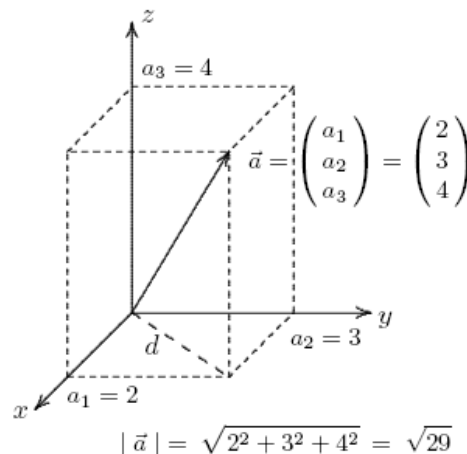


Abbildung 1: Betrag des Vektors (2 3 4)

Dank der Abbildung1 können wir einsehen, was für eine unkomplizierte Berechnung das Programm ausführen muss.

Im Eingabeaufforderungsfenster soll die Programmausführung so aussehen:

```

C:\
Bitte geben Sie die Komponenten des Vektors ein
Komponente 0: 2.0
Komponente 1: 3.0
Komponente 2: 4.0
Die laenge des Vektors ist: 5.38516
  
```

Abbildung 2: Beispiel der Ausführung des Programms

Dies muss der unterliegenden Quellcodevorlage zufolge implementiert werden, wo die Lücken und die farbigen Bereiche mit passenden Anweisungen auszufüllen sind.

```

/* NOTWENDIGE INCLUDES
   und ANDERE ANWEISUNGEN */

// DEKLARATION DER FUNKTIONEN betrag und quadrat1

int main(void)
{
    unsigned i;
    unsigned dim = 3; //Dimension des Vektors
    _____ // Deklaration eines Zeigers auf float für die Adresse des Vektors
    _____ // Deklaration einer float-Variable für den Betrag

    //Dynamische Reservierung eines Arrays mit "dim" Plätzen.
    vekt = _____;

    cout << "Bitte geben Sie die Komponenten des Vektors ein" << endl;

    _____ //Schleife Anweisung
    {
        cout << "Komponente " << i << ": ";
        _____ //Einlesen der Komponenten
    }

    //Aufruf der Funktion betrag
    _____ = betrag ( _____, _____ );

    //Freigabe des reservierten Speicherplatzes
    _____

    //Ausgabe der Vektorlänge
    cout << "\nDie laenge des Vektors ist: " << laenge << endl;

    return 0;
}

/* DEFINITION (Rümpfe) DER FUNKTIONEN
   betrag und quadrat1 */
_____ quadrat1 ( _____ )
{
    return _____;
}
_____ betrag ( _____, _____ )
{
}

```

Die Funktionen:**–quadrat1**

- **Übergabeparameter:** eine dezimale Zahl.
- **Rückgabe:** eine dezimale Zahl
- **Rumpf:** Die übergebene Zahl wird quadriert und zurückgegeben.

–betrag

- **Übergabeparameter:** eine Adresse (für den Array) und eine ganzzahlige Zahl (für die Dimension des Vektors)

- **Rückgabe:** eine dezimale Variable.
- **Rumpf:** Der Betrag des Vektors wird berechnet nach der Formel $v_1^2 + v_2^2 + v_3^2$ mittels einer Schleife. **Das Quadrat wird anhand der Funktion *quadrat1* berechnet.** Anschließend wird die Wurzel davon gerechnet und das Ergebnis zurückgegeben.

AUFGABE 1: Ersetzen Sie die Funktion *quadrat1* in vorherigen Programm, die *per Wert* aufgerufen wird, durch die Funktion *quadrat2*, die *per Referenz* aufgerufen wird.

– *quadrat2*

- **Übergabeparameter:** ein Zeiger auf eine Variable für dezimale Zahl.
- **Rückgabe:** Keine!
- **Rumpf:** Der Wert in der übergebene Adresse wird quadriert. Keine Rückgabe notwendig.

AUFGABE 2: **Ohne das Programm auszuführen** finden Sie die Ausgabe des Programms heraus.

```
#include <iostream.h>
#include <stdlib.h>
using namespace std;
//Deklaration der Funktion
void funkt (int *p);

int main(void)
{
    //Deklaration des Arrays
    int array[5] = { -1, -2, -3, -4, -5};

    //Aufruf der Funktion mit array als Übergabeparameter.
    funkt ( array );

    //Ausgabe von bestimmten Arrayelementen
    cout << array[0] << " ";
    cout << array[1] << " ";
    cout << array[2] << endl;

    return 0;
}

//Definition der Funktion
void funkt (int *p)
{
    p[0] = 3;  p[2] = -p[1];
}
```

AUFGABE 3: **Ohne die Programme auszuführen** finden Sie die Ausgabe beider folgenden Programme heraus. Was sind die Unterschiede zwischen beiden Programmen?

Text 1: Programm A

```
#include <iostream>
#include <stdlib.h>
using namespace std;

//Deklaration der Funktion
int summe (int , int );

int main(void)
{
    //Deklarationen
    int x = 2, y = -5;
    int z;

    //Aufruf der Funktion
    z = summe ( x, y );

    cout << "Die Summe ist: " << z << endl;

    return 0;
}

//Definition der Funktion
int summe (int a, int b)
{
    return a + b;
}
```

Text 2: Programm B

```
#include <iostream>
#include <stdlib.h>
using namespace std;

//Deklaration der Funktion
void summe (int , int, int *);

int main(void)
{
    //Deklarationen
    int x = 2, y = -5;
    int z;

    //Aufruf der Funktion
    summe ( x, y, &z );

    cout << "Die Summe ist: " << z << endl;

    return 0;
}

//Definition der Funktion
void summe (int a, int b, int *c)
{
    *c = a + b;
}
```