

Grundlagen der Informatik II – 4. Übung

Schwerpunkt: Wiederholungsfragen, dynamische Speicherverwaltung mit „new“ und „delete“

Wiederholungsfragen

1. Warum verwendet man überhaupt Verweise (Referenzen oder Zeigervariablen)?
2. Welche alltäglichen Analogien finden Sie zu Verweisen?
3. Was ist der signifikante Unterschied zwischen Referenzen und Zeigervariablen?
4. Wie definiert man Referenzen und wie definiert man Zeigervariablen?
5. Wie erreicht man das Objekt, auf das mit Hilfe von Referenzen oder Zeigervariablen gezeigt wird?
6. Wie bildet man eine Zeigervariable auf ein existierendes Feld?
7. Wie greift man über eine Zeigervariable auf die Elemente eines Feldes zu?

Dynamische Speicherverwaltung mit „new“ und „delete“

- Ziel: während der Laufzeit Speicher erzeugen und freigeben
- um neu erzeugten *Speicher* zu *adressieren*, werden *Zeigervariablen* verwendet

Dynamische Speicherreservierung für elementare Datentypen

- Speicher erzeugen: `datentyp *zeiger = new datentyp;`
- Speicher freigeben: `delete zeiger;`
- nach der Freigabe kann `zeiger` für andere Zwecke verwendet werden

Dynamische Speicherreservierung für Felder

- Speicherfeld erzeugen: `datentyp *zeiger = new datentyp[anzahl];`
- Speicherfeld freigeben: `delete[] zeiger;`
- nach der Freigabe kann `zeiger` für andere Zwecke verwendet werden

Beispielprogramm

```
#include <iostream>
using namespace std;

string* copystring(const string *s)
{
    // Speicher erzeugen
    string *ptr_s = new string;
    // Wert in neuen Speicher schreiben
    *ptr_s = *s;
    // Zeiger auf Speicher zurückgeben
    return ptr_s;
}

int* copyfeld(const int *f, int len)
{
    // Feldspeicher erzeugen
    int *ptr_f = new int[len];
    // Elemente einzeln eintragen
    for(int i=0; i<len; i++)
        ptr_f[i] = f[i];
    // Zeiger auf Speicherfeld zurückgeben
    return ptr_f;
}

int main()
{
    // Originale Objekte
    string msg = "Ich bin eine Zeichenkette.";
    int feld[] = { 1, 2, 3, 4, 5 };

    // Kopie der Objekte erzeugen
    // feld == &feld[0] ... siehe letzte Übung
    string *copy_msg = copystring(&msg);
    int *copy_feld = copyfeld(feld,5);

    // Objekte ausgeben
    cout << *copy_msg << endl;
    for(int i=0; i<5; i++)
        cout << copy_feld[i] << " ";

    // Speicher freigeben, wenn nicht mehr benötigt
    delete copy_msg;
    delete[] copy_feld;

    return 0;
}
```