# Designentscheidungen

Bei den ADTs für die drei Kategorien Mitarbeiter, Abteilungen und Abteilungsleiter, habe ich mich dafür entschieden die Verknüpfung durch Referenzen zu realisieren. Ich habe bei Mitarbeitern ein Integer eingefügt, in dem die Index Zahl der zugehörigen Abteilung gespeichert wird. Somit ist jeder Mitarbeiter mit allen gespeicherten Informationen der Abteilung verlinkt. Durch ein Integer im Abteilungsdatentyp, wird die Index Zahl des Leiters gespeichert. Somit sind alle drei Datentypen miteinander verknüpft und es können sich über den Mitarbeiter alle relevanten Informationen ausgeben lassen. Ich habe versucht zu vermeiden den Mitarbeiterdatentyp in dem Datentyp des Abteilungsleiters zu integrieren. Dadurch gibt es zwei getrennte Datenbanken und diese können auch getrennt ausgegeben und gespeichert werden.

Des Weiteren habe ich die Abteilung als Grundlage benutzt. Um einen Mitarbeiter oder einen Abteilungsleiter einzugeben muss zuerst eine Abteilung existieren. Damit konnte ich mein Programm so einrichten, dass der Benutzer die Abteilung aus einer Liste auswählen kann. Gleichzeitig verhindere ich eine Konsoleneingabe der Abteilung, denn diese ist schwer zu kontrollieren und mit den anderen Informationen zu vergleichen.

Das bedeutet aber auch, dass eine Abteilung existieren kann, ohne dass sie einen Abteilungsleiter besitzt. Für den Fall gibt es bei der Ausgabe einer Abteilungsliste (genauso bei der Mitarbeiterliste) eine Abfrage die prüft ob ein Abteilungsleiter besteht. Wenn das nicht der Fall ist, wird nur die Abkürzung TBD (to be decided) ausgegeben. Ein weiterer Grund ist, dass das Löschen eines Abteilungsleiters möglich sein soll. Ohne die Prüfung ob ein Leiter eingetragen ist, könnten dann nicht definierte Ausgaben erfolgen.

Abteilungsleiter werden in meinem Programm automatisch in die Liste der Mitarbeiter aufgenommen. Mir erschien es logisch die Abteilungsleiter auch als Mitarbeiter zu speichern, denn trotz Führungsposition sind sie Mitarbeiter der Firma. Bei der Ausgabe der Mitarbeiterliste hat man somit alle Personen die eingestellt sind auf einer Liste.

Das Löschen und Ändern von Einträgen habe ich in die Suchfunktion integriert. So kann der Benutzer den gewünschten Eintrag suchen anhand seiner gegebenen Informationen und diesen schnell entfernen oder anpassen. Zudem funktioniert eine Änderung oder ein Entfernen des Eintrags immer mit der jeweiligen Index Nummer. Diese ist dem Nutzer natürlich nicht bekannt, somit ist das Löschen über die Suchfunktion eine deutlich simplere Methode.

Personalnummern der Mitarbeiter werden in meinem Programm automatisch vergeben. Das hat den Vorteil das der Benutzer keine Fehler machen kann, durch falsche Eingaben. Dazu werden Personalnummern beibehalten, auch wenn Einträge gelöscht werden. So behält jeder Mitarbeiter seine eigene Nummer und es müssen in den Unterlagen etc. keine großen Änderungen vorgenommen werden. In dem speziellen Fall, dass der neueste Mitarbeiter gelöscht wird, wird seine Personalnummer auch gelöscht und zurückgesetzt. Dabei werden aber keine Nummern verschoben und es wird nichts verkompliziert.

Bei der Eingabe der Adresse habe ich mich dafür entschieden die Hausnummer als String zu speichern. Damit sind Hausnummern mit Buchstabenendungen möglich. Ohne diese Endungen käme es vermutlich zu Problemen bei Postzustellungen oder Ähnlichem. Die Postleitzahl habe ich begrenzt von 1001 – 99998. Nach kurzer Recherche im Internet ergab sich, dass die deutschen PLZ sich in diesem Fenster bewegen und somit machen andere Zahlen keinen Sinn.

Die Eingabe des Eintrittsdatums habe ich zuerst über die Eingabe von Integern realisiert. Damit fiel die Kontrolle der Eingabe leichter. So prüfe ich in dem Programm ob die Jahreszahl zwischen 1900 und 2100 liegt, für alle Zahlen außerhalb dieses Bereichs ist das Programm nicht ausgelegt. Genauso der Monat, der zwischen 1 und 12 liegen muss, aus ersichtlichen Gründen. Danach folgt die Konvertierung in einen String um eine Speicherung in dem MM/JJJJ Format zu realisieren.

Telefonnummern sind bei ebenfalls als String eingetragen. Dies hat mehrere Gründe. Erstens sind Telefonnummern nicht sicher in dem Bereich den der Integer erfassen würde. Zweiten ist so eine bessere Formatierung möglich, um auch Vorwahlen zu speichern (z.B. 040 würde zu 40 werden und somit jeglichen Nutzen verlieren). Und Zuletzt gibt es noch die Möglichkeit Ländervorwahlen mit einzugeben (z.B. +49 für Deutschland).

Bei den E-Mails lasse ich alle Leerzeichen durch Unterstriche ersetzen. Leerzeichen in E-Mails sind nicht zugelassen. Großbuchstaben werden genauso durch Kleinbuchstaben ersetzt (Datensätze sind noch mit Großbuchstaben gespeichert, da sonst alle Datensätze neu eingegeben werden müssten). Dies hat nur den Grund um die Adressen einheitlicher zu gestalten.

Zuletzt habe ich mich dafür entschieden, das Programm beim Beenden nicht zu speichern. Der Benutzer muss die dafür vorgesehene Option benutzen. Das hat den Grund, dass der Benutzer im Falle eines Fehlers das Programm einfach beenden kann und mit seinen alten Daten weiterarbeiten kann, ohne das etwas gelöscht wird.