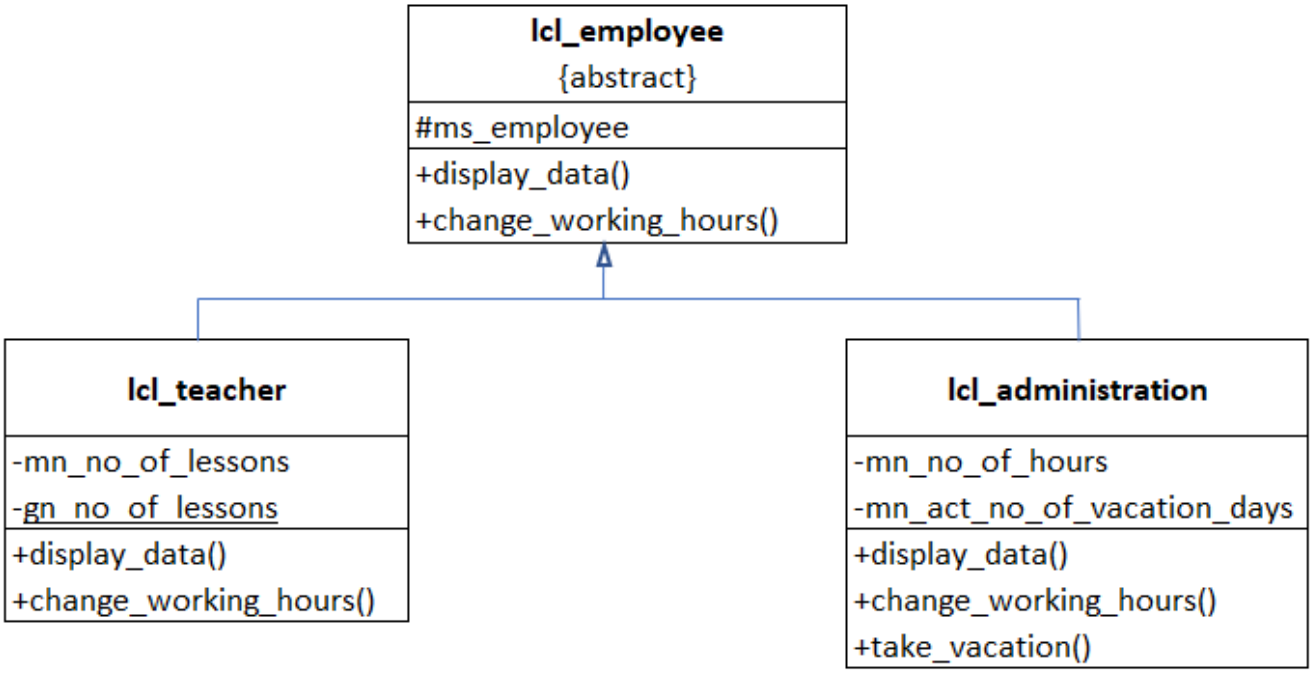
Szenario:

Die allgemeine Klasse Employee umfasst alle Angestellten der Schule. Dazu gehören Lehrer und Angestellte in der Administration. Weitere Angestellte gibt es nicht.

Als Grundlage dient das folgende Klassendiagramm. Dafür sollen nun schrittweise die benötigten Klassen angelegt werden.



Aufgabe 1 – Klasse Employee

a) Legen Sie eine abstrakte Klasse Employee an (ZXXX_XX_CL_EMPLOYEE).
b) Jeder Angestellte der Schule hat einen Namen, einen Vornamen, eine Büronummer und ein Kürzel. Dafür wird der Typ ts_employee in der Klasse Employee angelegt und umfasst folgende Komponenten:
   - `surname` (char Länge 30)
   - `name` (char Länge 20)
   - `office_nr` (int)
   - `kuerzel` (char Länge 2)

   Legen Sie anschließend das Attribut ms_employee an.
c) Legen Sie den Konstruktor an und übergeben eine Struktur vom Typ ts_employee und weisen diese dem Instanz-Attribut zu.
d) Alle Angestellten können ihre Daten abrufen uns ausgeben. Mit der Methode DISPLAY_DATA() werden alle Informationen mit Hilfe von WRITE ausgegeben:
   `surname name`
   Büro Nr. `office_nr` (`kuerzel`)
e) Jeder Angestellte der Schule kann seine Arbeitsstunden ändern. Die dafür benötigte abstrakte Methode CHANGE_WORKING_HOURS hat einen Parameter IN_DIFFERENCE vom Typ I.

Aufgabe 2 – Klasse Teacher

a) Legen Sie eine Public Klasse Teacher an (ZXXX_XX_CL_TEACHER) und Kennzeichen diese als Final.
b) Tragen Sie ZXXX_XX_CL_EMPLOYEE als Oberklasse ein.
c) Lehrer haben eine Normalarbeitszeit von 24 Unterrichtsstunden. Legen Sie dafür ein privates statisches Attribut GN_NO_OF_LESSONS an und setzen den Wert auf 24.
d) Es ist aber möglich, dass Lehrer weniger Stunden arbeiten. Die tatsächliche Arbeitszeit wird in einem weiteren Attribut MN_NO_OF_LESSONS gespeichert.
e) Legen Sie den Konstruktor an und übergeben zusätzlich zur Struktur noch die Anzahl der tatsächlichen Stunden (IN_NO_OF_LESSONS).
f) Die Methode CHANGE_WORKING_HOURS muss redefiniert werden, da sie in der Oberklasse als abstrakt gekennzeichnet ist. Jeder Lehrer muss mindestens 2 Stunden und darf maximal 27 Stunden arbeiten. Größere Abweichungen sind nicht erlaubt. Eine Fehlerprüfung muss durchgeführt werden und mit Hilfe von WRITE einer der beiden Meldungen ausgegeben werden:
    - „Änderung der Stundenzahl nicht möglich!“
    - „Ändere Stundenzahl ...“
g) Die Methode DISPLAY_DATA erweitert die Ausgabe von der Oberklasse um die Information zur aktuellen Stundenzahl.

Aufgabe 3 – Hauptprogramm zum Testen der Klasse Teacher

a) Legen Sie ein Programm ZXXX_XX_VERERBUNG_EMPLOYEE an und testen ihre Klasse Teacher und Employee.
b) Legen Sie eine Struktur vom Typ ts_employee an und befüllen diese mit beliebigen Werten.
c) Erzeugen Sie ein neues Objekt ihrer Klasse Teacher und speichern dieses in einer Referenz an.
d) Rufen Sie die Methode DISPLAY_DATA auf.
e) Rufen Sie die Methode CHANGE_WORKING_HOURS auf.
f) Rufen Sie die Methode DISPLAY_DATA erneut auf.

Aufgabe 4 – Klasse Administraton

a) Legen Sie eine Public Klasse Administraton an (ZXXX_XX_CL_ADMINISTRATION) und Kennzeichen diese als Final.
b) Tragen Sie ZXXX_XX_CL_EMPLOYEE als Oberklasse ein.
c) Angestellte in der Administration haben eine Normalarbeitszeit von 40 Stunden. Angestellte können außerdem Urlaub nehmen. Daher sind die aktuell freien Urlaubstage als Attribut zu speichern.
   Erweitern Sie die Klasse um zwei neue Instanz Attribute MN_NO_OF_HOURS und MN_ACT_NO_OF_VACATION_DAYS. Belegen Sie die Stunden mit den Wert 40 vor und die Urlaubstage mit den Wert 30.
d) Legen Sie den Konstruktor an und übergeben zusätzlich zur Struktur noch die Anzahl der aktuell freien Urlaubstage (IN_ACT_NO_OF_VACATION_DAYS). Ist die übergebene Anzahl nicht negativ und kleiner gleich 40, dann wird diese im zugehörigen Instanz-Attribut gespeichert. Ansonsten wird der ursprüngliche Wert von 30 nicht verändert.
   Für die Normalarbeitszeit wird beim Konstruktor kein weiterer Parameter benötigt, da diese vorerst immer den Wert 40 hat.
e) Angestellte können ihre Arbeitszeit um ± 5 Stunden verändern. Die Methode CHANGE_WORKING_HOURS muss dafür redefiniert werden. Falls der Wert des übergebenen Parameters kleiner als -5 oder größer als $+5$ ist wird die Meldung „Änderung der Stundenzahl nicht möglich!“. Ansonsten wird die Stundenzahl entsprechend abgeändert und die Meldung „Ändere Stundenzahl ...“ ausgegeben.
   (Hinweis: Ausgaben jeweils mit WRITE)
f) Die Methode DISPLAY_DATA erweitert die Ausgabe von der Oberklasse um die Information zur aktuellen Arbeitszeit sowie zu den aktuell freien Urlaubstagen.
   Aktuelle Stundenzahl: 40
   Aktuelle Urlaubstage: 16
g) Angestellte können außerdem Urlaub nehmen. Dafür steht zusätzlich in dieser Klasse noch die Instanz Methode TAKE_VACATION zur Verfügung, der die Anzahl der gewünschten Urlaubstage übergeben werden (IN_NO_OF_DAYS). Ist die übergebene Anzahl kleiner als 0 wird die Meldung „Fehler: keine negative Anzahl an Tage möglich“ ausgegeben und die Methode mit RETURN verlassen.
   Weiter muss geprüft werden, ob noch genügend Urlaubstage zur Verfügung stehen. Wenn ja, dann werden diese von der aktuellen Anzahl abgezogen und die Meldung „Ändere Urlaubstage...“ ausgegeben. Ansonsten wird „zu wenig Urlaubstage vorhanden“ ausgegeben und keine Änderung der Urlaubstage vorgenommen.

Aufgabe 5 – Hauptprogramm zum Testen der Klasse Administraton

a) Erweitern Sie ihr Programm ZXXX_XX_VERERBUNG_EMPLOYEE (löschen oder ändern sie aber keinen bestehenden Code).
b) Erzeugen Sie ein neues Objekt ihrer Klasse Administraton. Die Struktur gs_employee kann dafür erneut verwendet werden und weisen Sie dieser die folgenden Werte zu:
   `surname = 'Meier', name = 'Gudrun', office_nr = 221, kuerzel = 'me'`
   Der Angestellt soll noch 14 Urlaubstage haben.
c) Rufen Sie die Methode DISPLAY_DATA auf.
d) Rufen Sie die Methode TAKE_VACATION und CHANGE_WORKING_HOURS auf.
e) Rufen Sie die Methode DISPLAY_DATA erneut auf.

Aufgabe 6 – Polymorphie

a) Kopieren Sie das Programm ZXXX_XX_VERERBUNG_EMPLOYEE nach ZXXX_XX_VERERBUNG_POLYMORPHIE
b) Löschen Sie die Ausgabe am Ende (d.h. den Aufruf der Methoden).
c) Legen Sie eine interne Tabelle gt_employees an, in der Sie Objekte der Klasse Employee speichern können.
d) Fügen Sie die beiden Objekte der Klasse Teacher bzw. Administration zur internen Tabelle hinzu.
e) Geben Sie in einer Schleife alles Angestellten aus und rufen Sie für jeden Angestellten die Methode CHANGE_WORKING_HOURS auf.