Friedrich-Schiller-Universität Jena **Fakultät für Mathematik und Informatik** **Institut für Informatik** **Lehrstuhl für Programmiersprachen und Compiler**	**Höhere Programmierung** **SS 2001**	**Aufgabenblatt 12** **Ausgabe: 18.06.2001** **Abgabetermin:** **25.06.2001 16:00 Uhr**

Aufgabe 1: Funktion für die robuste Eingabe

Entwickeln Sie die Funktion `RobusteEingabeFloat`, die einen Wert vom Typ Float einliest. Die Funktion gibt den eingelesenen Wert zurück.. Für die Reaktion im Falle fehlerhafter Eingabe soll eine der beiden folgenden Strategien realisiert werden:

- mit der Eingabe von "ende" soll die Funktion die Ausnahme `Eingabe_Ende` auslösen.
- Einlesen wiederholen, bis die Eingabe zulässig ist.

Entwickeln sie ein Programm, das die Funktion `RobusteEingabeFloat` testet.

4 Punkte

Aufgabe 2: Funktionen in arithmetischen Ausdrücken

Welches Ergebnis gibt das folgende Programm aus? Begründen Sie ihre Antwort.

```
with Ada.Text_IO;

PROCEDURE Berechnung IS
  X: Integer := 1;
  Y: integer := 2;

  FUNKTION Fkt1 RETURN Integer IS
  BEGIN
    X:=X+1;
    Y:=X*3;
    RETURN (Y*X);
  END Fkt1;

  FUNKTION Fkt2 RETURN Integer IS
  BEGIN
    X:=X*2;
    Y:=Y+X;
    RETURN (X+Y);
  END;

BEGIN
  Y:=Y + Fkt1 + X*Fkt2;
  Ada.Text_IO.Put_Line(integer'Image(Y));

END Berechnung;
```

4 Punkte

Aufgabe 3: Binomialkoeffizient

Entwickeln Sie eine Funktion, die den Binomialkoeffizienten $\binom{n}{k}$, mit n, k ∈ Natural und n ≥ k für gegebene n und k berechnet. Die Funktion soll für einen möglichst großen Wertebereich von n und k das mathematische korrekte Ergebnis liefern. Sonst ist eine geeignete Ausnahme auszulösen.

7 Punkte

Aufgabe 4: Abwandern eines binären Baumes (Zusatzaufgabe für Wirtschaftsinformatiker)

Ein binärer Baum, dessen Elemente einen String als Wert enthalten, soll nach einem bestimmten vom Benutzer eingegebenen Wert abgesucht werden. Entwickeln Sie ein Programm mit einer Funktion, die den Baum nach einem als Parameter übergebenen Stringwert absucht und die Anzahl der Treffer zurückgibt. Folgende Typ- und Variablendefinitionen seien bereits gegeben.

```
TYPE StringAccessType IS ACCESS String;

TYPE BaumKnotenType;
TYPE BaumAccessType IS ACCESS BaumKnotenType;
TYPE BaumKnotenType IS RECORD
   Wert: StringAccessType;
   VerweisLinkerTeilbaum: BaumAccessType;
   VerweisRechterTeilbaum: BaumAccessType;
END RECORD;

Wurzel: BaumAccessType;
```

Als Beispiel initialisieren Sie den Baum mit nichttrivialen Werten und mindestens 8 Knoten und einer maximalen Baumhöhe von 4. Dokumentieren Sie einen Testlauf mit einem im Baum zu findenden Stringwert.

5 Punkte