Oberon - Test1

[1] 20t

Kaj se izpiše med izvajanjem ukaza Vaja.Proc2 ?

```
MODULE Vaja;
IMPORT Out;
VAR d: INTEGER;

PROCEDURE Proc1(VAR c: INTEGER; a: INTEGER);
VAR c: INETGER;

BEGIN
  a:=a+2;
  b:=a+c;
  d:=c-a-1;
  Out.Int(a, 4); Out.Int(b, 4); Out.Int(c, 4); Out.Int(d, 4);
  Out.Ln;
END Proc1;

PROCEDURE Proc2*;
VAR a, b, c: INTEGER;

BEGIN
  a:=3; b:=4; c:=2;
  Proc1(c, a);
  Out.Int(a, 4); Out.Int(b, 4); Out.Int(c, 4); Out.Int(d, 4);
  Out.Ln;
END Proc2;

BEGIN
  d:=6;
END Vaja.
```

[2] 20t

Napišite podprogram ***MaxDeterminanta***, ki v neki matriki ***Mat*** (poljubno veliki dvodimenzionalni tabeli realnih števil) pogleda vse podmatrike reda 2 in izpiše tisto, ki ima največjo vrednost determinante:

```
PROCEDURE MaxDeterminanta(VAR Mat: ARRAY OF ARRAY OF REAL);
```

Izračun determinante: $\begin{vmatrix} x11 & x12 \\ x21 & x22 \end{vmatrix} = x11*x22 - x12*x21$

[3] 30t

Napišite podprogram za izračun povprečne ocene nekega študenta.
(a) Predpostavite da podatki o študentu ustrezajo naslednjim deklaracijam:

```
    TYPE Student = RECORD
      priimek, ime: ARRAY 20 OF CHAR;
      StOcen: INTEGER;
      Ocene: ARRAY 40 OF SHORTINT;
    END;

PROCEDURE PovpOcena(s: Student): REAL;
```

(b) Podana je tabela t, ki vsebuje podatke o študentih:

```
TYPE Tabela: ARRAY 100 OF Student;
VAR t: Tabela;
```

Napišite podprogram, ki uredi (sortira) tabelo t v padajočem vrstnem redu glede na povprečno oceno. Kot sestavni del rešitve uporabite funkcijski podprogram PovOcena iz 3a).

[4] 20t

Napišite podprogram, ki iz podatkov linearnega seznama zgradi množico. Preden nek podatek vstavite v množico preverite, ali je njegova vrednost v mejah, ki jih oberon dovoljuje za elemente množice. Podatke, ki ne izpolnjujejo tega pogoja zavrži. Upoštevajte naslednje deklaracije:

```
TYPE Kazalec = POINTER TO ElementSeznama;
  ElementSeznama = RECORD
         Podatek: INTEGER;
         Vezava: Kazalec;
  END;

PROCEDURE IzSeznamaVMnožico(Zac: Kazalec): SET;
```

[5] 20t

Napišite generični modul SkladiG, ki bo omogočal realizacijo sklada (LIFO) s podatki poljubnega tipa (dejanski tip podatkov je deklariran kasneje s pomočjo ustrezne rešitve tipa v modulu odjemalcu).
Generični modul naj vsebuje podprogram ***Push*** (dodajanje elementa na vrh sklada), ***Pop*** (odvzemanje iz vrha sklada), ***Open*** (inicializacija sklada) in ***Empty*** (test, ali je sklad prazen)) v skladu z naslednjo definicijo:

```
DEFINITION SkladiG;
  TYPE Kazalec = PPOINTER TO Element;
  Element = RECORD END;
  Sklad = RECORD END;

PROCEDURE  Push(VAR s: Sklad; novi: Kazalec);
           Pop(VAR s: Sklad): Kazalec;
           Open(VAR s: Sklad);
           Empty(s: Sklad): BOOLEAN;
END SkladG;
```