```
{ Dorim sa definim un tip de date pentru reprezentarea de
  figuri geometrice. Avem nevoie sa reprezentam trei tipuri
  de figuri geometrice:
   * cerc - se pastreaza raza;
   * triunghi - se pastreaza lungimile celor trei laturi;
   * dreptunghi - se pastreaza lungimea si latimea.

  Vom folosi tipul articol pentru a defini tipuri de date
  pentru figurile geometrice. Vom folosi atat articol fara
  variante, cat si articol cu variante, si vom compara spatiul
  ocupat in memorie de cele doua. }
program Spatiu_ocupat_de_tipul_articol_in_memorie;

type
    { O enumerare pentru tipurile de figuri geometrice. }
    Fig = (Cerc, Triunghi, Dreptunghi);

    { Definirea unui tip de date pentru figurile geometrice
      folosind articol *cu* variante. }
    ConturCuVariante = record
         Perimetru : real;
         case Fel : Fig of
              Cerc : (Raza : Integer);
              Triunghi : (Latura1, Latura2, Latura3 : Integer);
              Dreptunghi : (Lungime, Latime : Integer);
    end;

    { Definirea unui tip de date pentru figurile geometrice
      folosing articol *fara* variante. }
    ConturFaraVariante = record
         Perimetru : Real;
         Fel : Fig;
         Raza : Integer;
         Latura1, Latura2, Latura3 : Integer;
         Lungime, Latime : Integer;
    end;

{ Definim cate o variabila din fiecare din cele doua tipuri
  definite. }
var
   ccv : ConturCuVariante;
   cfv : ConturFaraVariante;

begin
     { Pentru inceput vrem sa vedem cat ocupa in memorie
       tipurile elementare de date. }
     WriteLn('Dimensiune Integer: ', SizeOf(Integer));
     WriteLn('Dimensiune Real: ', SizeOf(Real));
     WriteLn('Dimensiune enumerare Fig: ', SizeOf(Fig));
     WriteLn;

     { Vrem sa vedem cat ocupa in memorie fiecare din cele
       doua tipuri de date pe care le-am definit.
       Vom vedea ca articolul cu variante ocupa mai putin spatiu. }
     WriteLn('Dimensiune articol cu variante: ', SizeOf(ConturCuVariante));
     WriteLn('Dimensiune articol fara variante: ', SizeOf(ConturFaraVariante));

     { Vrem sa vedem cat castigam (in procente) daca folosim
       articol cu variante fata de articol fara variante. }
     WriteLn('Casting: ', 100 * (SizeOf(ConturFaraVariante) -
         SizeOf(ConturCuVariante)) / SizeOf(ConturFaraVariante):0:2, '%');
```

```
    WriteLn;

    { In principiu cele doua tipuri de date pe care le-am definit sunt
      echivalente. Daca le folosim cu grija, ele pot face exact acelasi lucru.

      O diferenta este spatiul ocupat in memorie. Dupa cum am vazut mai
      sus, articolul cu variante ocupa mai putin spatiu, deci este mai
      avantajos sa il folosim.

      O alta diferenta este faptul ca la articolul cu variante, campurile
      de la variante se suprapun in memorie. De fapt aceasta este si cauza
      pentru care se ocupa mai putin spatiu de memorie.

      Pentru exemplul nostru, campul de la varianta "Cerc" (adica "Raza")
      se suprapune cu campurile de la varianta "Triunghi" (adica "Latura1",
      "Latura2" si "Latura3") si se mai suprapune si cu campurile de la
      variante "Dreptunghi" (adica "Lungime" si "Latime"). Ca urmare daca
      modificam valoarea campului "Raza", implicit se va modifica si
      valoarea campului "Latura1" si a campului "Lungime".

      O asemenea suprapunere de campuri nu are loc la articolul fara
      variante. }

    { Initializam campul "Raza" si apoi campul "Latura1" in cazul
      articolului cu variante. Vom vedea ca dupa ce initializam
      campul "Latura1" se va modifica si campul "Raza" (din cauza
      suprapunerii de care am spus mai sus). }
    ccv.Raza := 10;
    ccv.Latura1 := 20;
    WriteLn('Raza contur cu variante: ', ccv.Raza);
    WriteLn('Latura1 contur cu variante: ', ccv.Latura1);
    WriteLn;

    { Facem aceleasi initializari in cazul articolului fara variante.
      Vom vedea ca, de data aceasta, dupa ce initializam campul "Latura1",
      campul "Raza" nu mai este afectat. }
    cfv.Raza := 10;
    cfv.Latura1 := 20;
    WriteLn('Raza contur fara variante: ', cfv.Raza);
    WriteLn('Latura1 contur fara variante: ', cfv.Latura1);
end.
```