Programa E.37 *Rotula grafo e atribui valores para o arranjo g*

```
void Atribuig (TipoGrafo *Grafo,
               TipoArranjoArestas L,
               Tipog g)
{ int i, j, u, Soma; TipoValorVertice v; TipoAresta a;
  unsigned char Visitado[MAXNUMVERTICES];
  for (i = Grafo->NumVertices - 1; i >= 0; i--)
    { g[i] = Grafo->r; Visitado[i] = FALSE; }
  for (i = Grafo->NumArestas - 1; i >= 0; i--)
    { a = L[i]; Soma = 0;
      for (v = Grafo->r - 1; v >= 0; v--)
        { if (!Visitado[a.Vertices[v]])
          { Visitado[a.Vertices[v]] = TRUE;
            u = a.Vertices[v]; j = v;
          }
          else Soma += g[a.Vertices[v]];
        }
      g[u] = (j - Soma) % Grafo->r;
      while (g[u] < 0) g[u] += Grafo->r;
    }
}
```

Programa E.38 *Função de transformação perfeita*

```
TipoIndice hp (TipoChave Chave,
               Tipor r,
               TipoTodosPesos Pesos,
               Tipog g)
{ int i, v = 0; TipoArranjoVertices a;
  for (i = 0; i < r; i++)
  { a[i] = h(Chave, Pesos[i]);
    v += g[a[i]];
  }
  v = v % r; return a[v];
}
```

Programa E.39 *Rotula grafo e atribui valores para o arranjo g usando 2* bits *por entrada*

```
/* Assume que todas as entradas de 2 bits do vetor */
/* g foram inicializadas com o valor 3             */
void AtribuiValor2Bits (Tipog *g,
                        int Indice,
                        unsigned char Valor)
{ int i, Pos;
  i   = Indice / 4;
  Pos = (Indice % 4);
  Pos = Pos * 2;          /* Cada valor ocupa 2 bits */
  g[i] &= ~(3U<< Pos);    /* zera os dois bits a atribuir */
  g[i] |= (Valor << Pos); /* realiza a atribuicao */
```


Continuação do Programa E.39

```
} /* AtribuiValor2Bits */

char ObtemValor2Bits (Tipog *g, int Indice)
{ int i, Pos;
  i = Indice / 4;
  Pos = (Indice % 4);
  Pos = Pos * 2; /* Cada valor ocupa 2 bits */
  return (g[i] >> Pos) & 3U;
} /* ObtemValor2Bits */

void Atribuig (TipoGrafo *Grafo,
               TipoArranjoArestas L,
               Tipog *g)
{ int i, j, u, Soma; TipoValorVertice v; TipoAresta a;
  unsigned int valorg2bits; unsigned char Visitado[MAXNUMVERTICES];
  if (Grafo->r <= 3)
  { /* valores de 2 bits requerem r <= 3 */
    for (i = Grafo->NumVertices - 1; i >= 0; i--)
    { AtribuiValor2Bits(g, i, Grafo->r);
      Visitado[i] = FALSE;
    }
    for (i = Grafo->NumArestas - 1; i >= 0; i--)
    { a = L[i]; Soma = 0;
      for (v = Grafo->r - 1; v >= 0; v--)
      { if (!Visitado[a.Vertices[v]])
        { Visitado[a.Vertices[v]] = TRUE;
          u = a.Vertices[v];
          j = v;
        }
        else Soma += ObtemValor2Bits(g, a.Vertices[v]);
      }
      valorg2bits = (j - Soma) % Grafo->r;
      while (valorg2bits > Grafo->r) valorg2bits += Grafo->r;
      AtribuiValor2Bits (g, u, valorg2bits);
    }
  }
} /*--Fim Atribuig--*/
```

Programa E.40 *Gera a tabela TabRank*

```
void GeraTabRank (Tipog *g, TipoValorVertice Tamg,
                  TipoK k, TipoTabRank *TabRank)
{ int i, Soma = 0;
  for (i = 0; i < Tamg; i++)
    { if (i % k == 0) TabRank[i / k] = Soma;
      if (ObtemValor2Bits(g, i) != NAOATRIBUIDO) Soma = Soma + 1;
    }
} /* GeraTabRank */
```