// Soluzione iterativa completa

```
void computeM(int*T, int*P, int dimT, int dimP, bool*M)
{

  for(int i=0; i<dimP; i++)
    for(int j=0; j<dimT; j++)
     {
          *(M+j*dimP+i)=P[i]==T[j];
     }
}
//POST= riempie M correttamente
int auxI(bool * M, int i, int j, int dimP, int dimT)
{
  int l=0;
  while(i<dimP && j <dimT && M[i+j*dimP])
    l++;
  return l;

}
//POST= trova max match che inizia da P[i] e T[j]

tripla match(bool*M,int dimP, int dimT)
{
  tripla best;
  for(int i=0; i < dimP&&(dimP-i)>best.lung; i++)
    {
      for(int j=0; j< dimT; j++)
          {
            int x= auxI(M, i, j, dimP, dimT);
            if(x > best.lung)
              best=tripla(j,i,x);
          }
    }
  return best;
}
//POST=ritorna best match
```

// soluzione ricorsiva che segue quella iterative: 3 cicli corrispondono a 3 funzioni ricorsive

```
int aux3(bool*M, int p, int t, int dimT, int dimP)
{
  if(p<dimP && t < dimT)
    {
      if(M[p+t*dimP])
          return 1 + aux3(M, p+1, t+1, dimT, dimP);
      else
          return 0;
    }
  return 0;
}
//POST=restituisce  massimo match che inizia in P[p] e in T[t]
```

```
tripla aux2(bool*M, int p, int t, int dimT, int dimP)
{
  if(t<dimT)
    {
      int lung=aux3(M,p,t,dimT,dimP);
      tripla x=aux2(M,p,t+1,dimT,dimP);
      if(lung>=x.lung)
            return tripla(t,p,lung);
      else
            return x;
    }
  return tripla();
}
//Restituisce la tripla che corrisponde al massimo match in T che inizia in P[P]

tripla matchR(bool*M, int p, int dimT, int dimP)
{
  if(p<dimP)
    {
      tripla y=aux2(M,p, 0, dimT, dimP);
      tripla x= matchR(M, p+1, dimT,dimP);
      if(x.lung > y.lung)
            return x;
      else
            return y;
    }
  else
    return tripla();
}
//POST= restituisce la tripla che corrisponde al massimo match di P in T
```