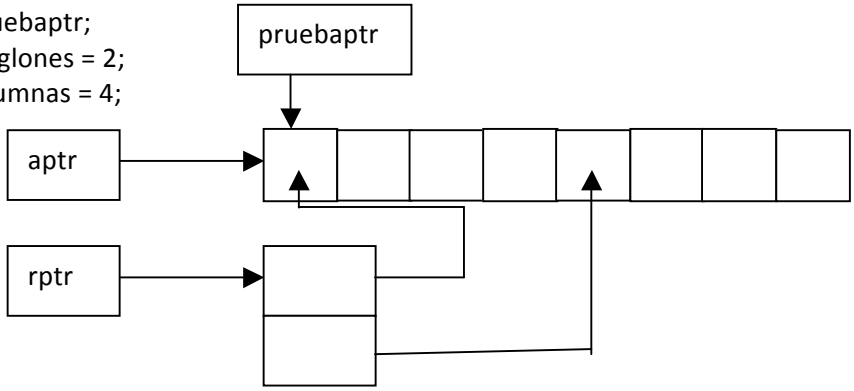
ARREGLO BIDIMENSIONAL DINÁMICO (Método 4, Tutorial pointersC.pdf)

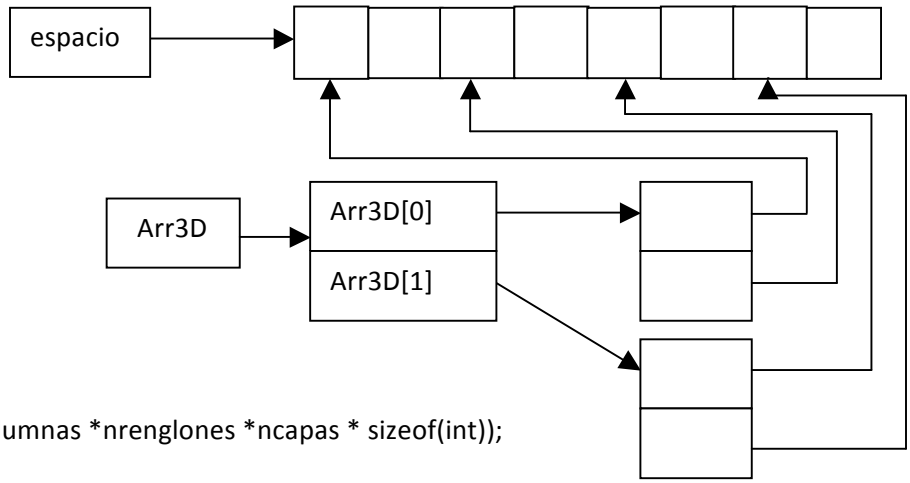
```
int **rptr;
 int *aptr;
 int *pruebaptr;
int nrenglones = 2;
int ncolumnas = 4;
```




```
aptr = malloc(nrenglones * ncolumnas * sizeof(int));
rptr = malloc(nrenglones * sizeof(int *));
for (k = 0; k < nrenglones; k++)
      rptr[k] = aptr + (k * ncolumnas);
```

ARREGLO TRIDIMENSIONAL (Método 4, pointersC.pdf)

```
int *espacio;
int ***Arr3D;
int ncapas=2, nrenglones = 2, int ncolumnas = 2;
```

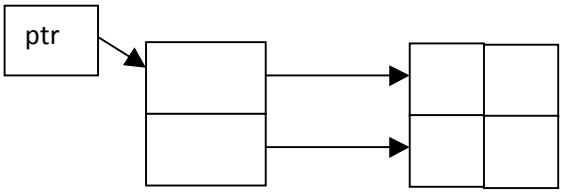



```
espacio = malloc(ncolumnas *nrenglones *ncapas * sizeof(int));

Arr3D = malloc(ncapas * sizeof(int **));
for (z = 0; z <ncapas; z++)
{
      Arr3D[z] = malloc(nrenglones * sizeof(int *));
     for (y = 0; y < nrenglones; y++)
          Arr3D[z][y] = espacio + (z*(ncolumnas *nrenglones) + y*ncolumnas);
```

}

ARREGLO BIDIMENSIONAL (ejemplo 9.2)

```
int **ptr;
int renglon;

 ptr = (int**) malloc(nrenglones * sizeof(int *));
  for (renglon = 0; renglon < nrenglones; renglon++)
    ptr[renglon] = (int*) malloc(ncolumnas * sizeof(int));
```



ARREGLO TRIDIMENSIONAL (ejemplo 9.2)

```
int ***ptr;
int capas,renglon;

 ptr = (int***) malloc(ncapas * sizeof(int **));
 for (capas = 0; capas < ncapas; capas++)
 {
  ptr[capas] = (int**) malloc(nrenglones * sizeof(int *));
  for (renglon = 0; renglon < nrenglones; renglon++)
      ptr[capas][renglon] = (int*) malloc(ncolumnas * sizeof(int));
 }
```