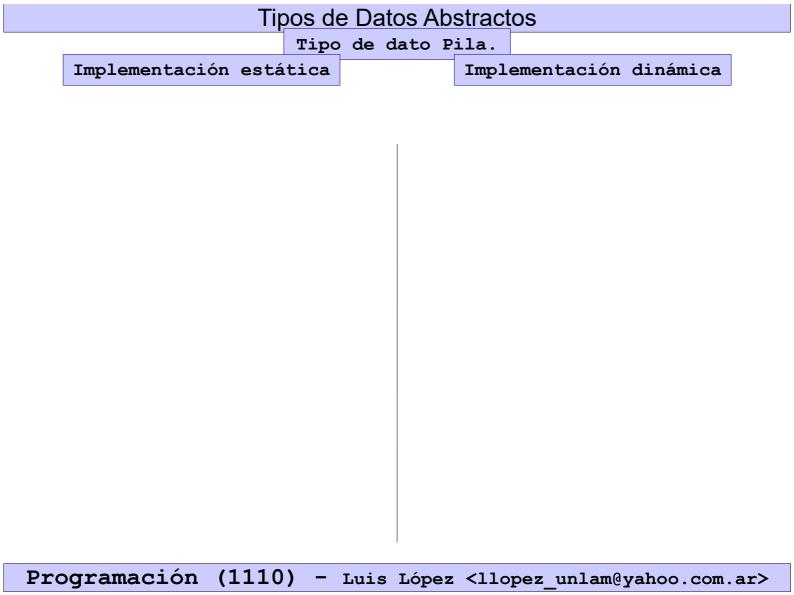
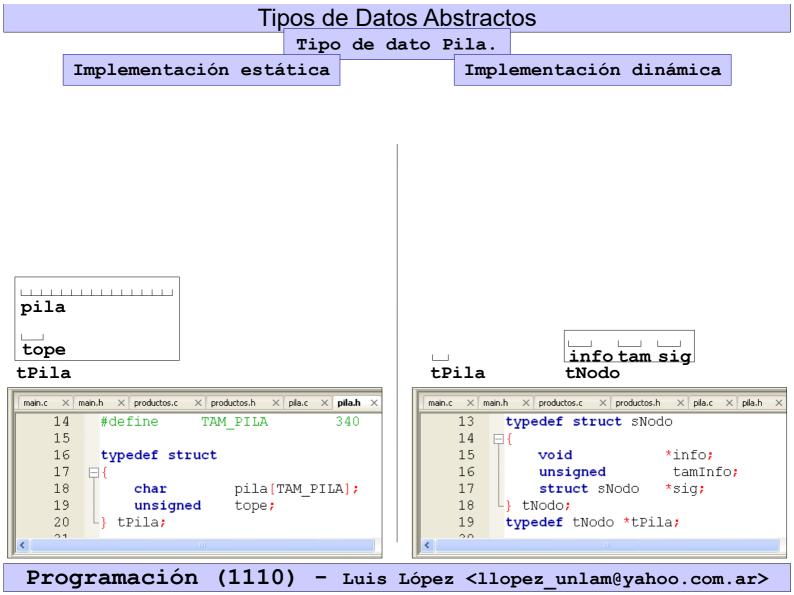
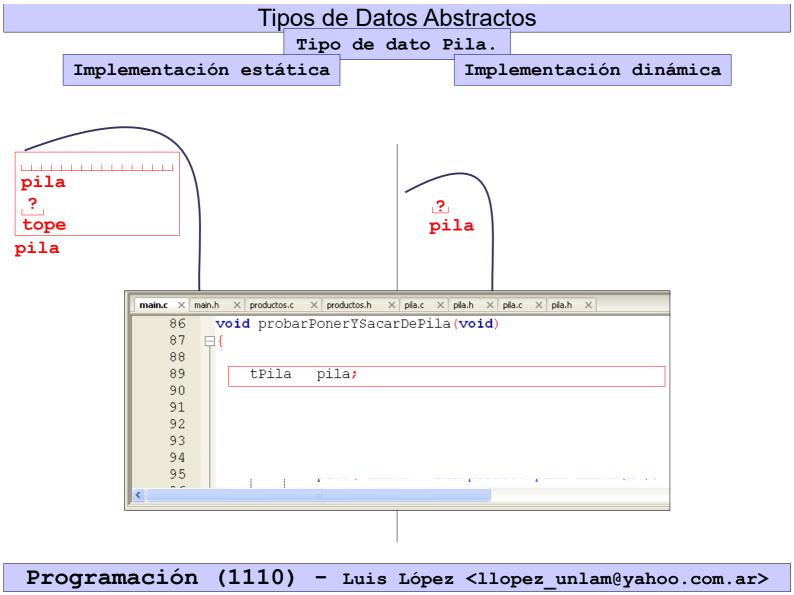


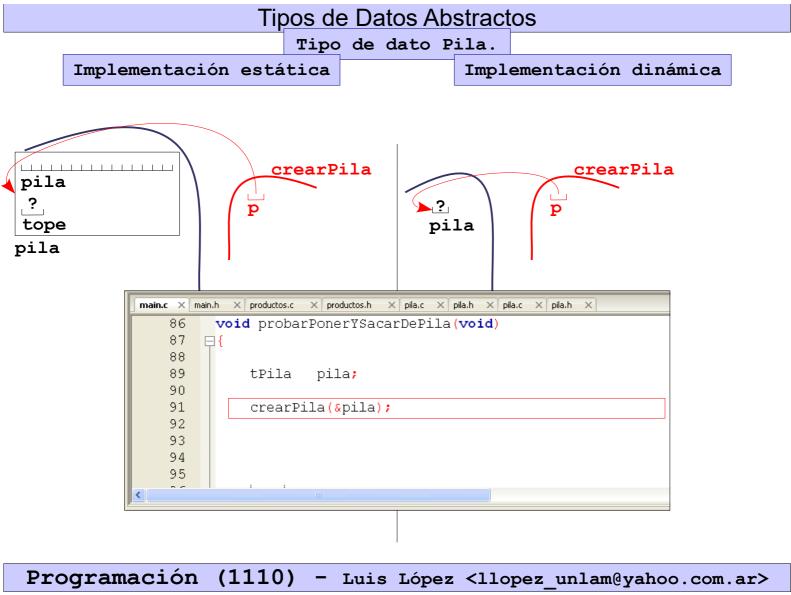
Programación (1110) - Luis López clopez_unlam@yahoo.com.ar>

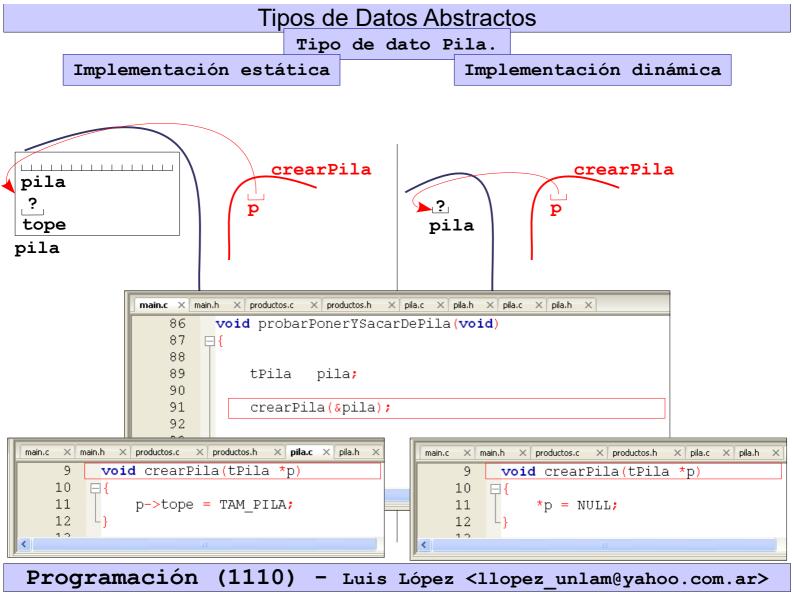
Tipos de Datos Abstractos
Tipo de dato Pila.

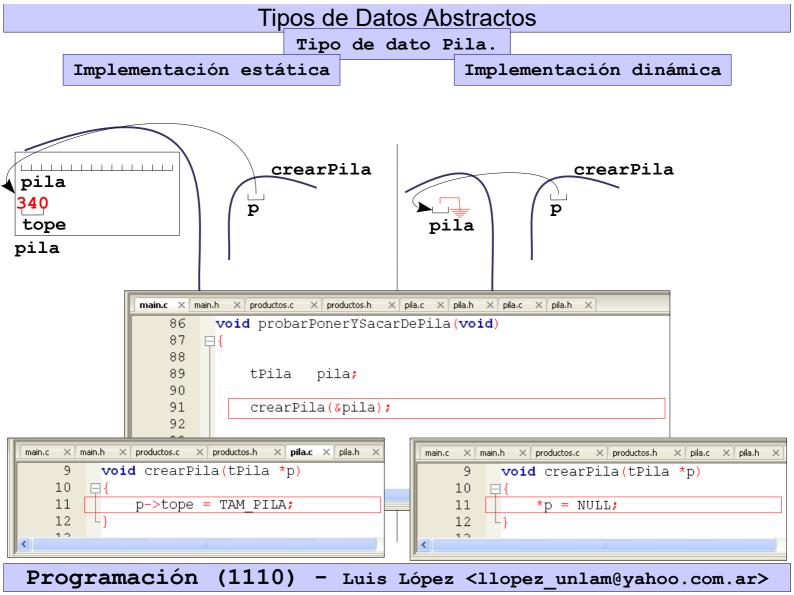


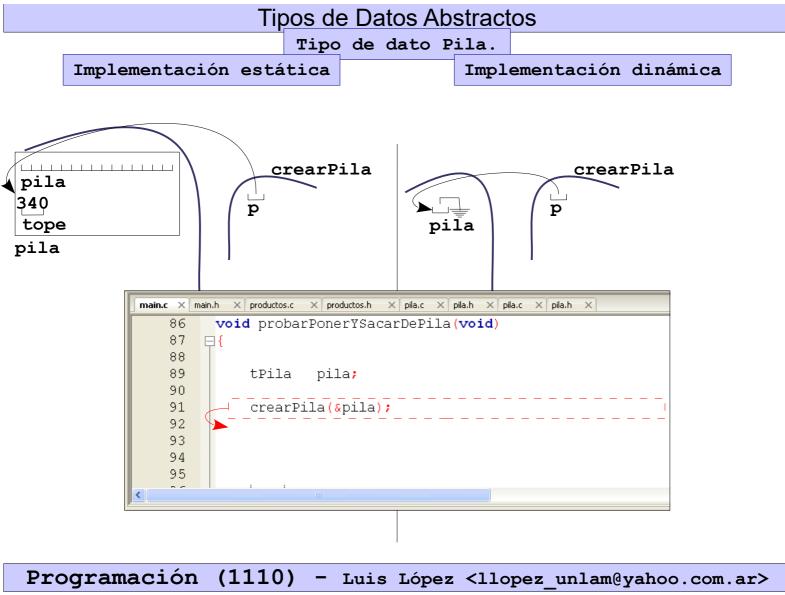




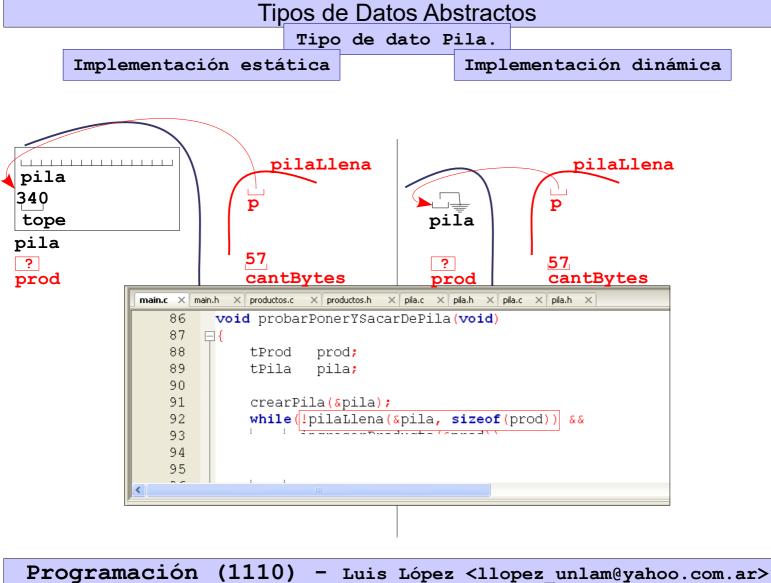












Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pilaLlena pilaLlena pila 340 tope pila pila **57** 57 ? ? cantBytes cantBytes prod prod X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h 86 void probarPonerYSacarDePila(void) 87 88 tProd prod; t.Pila pila; x main.h x productos.c x productos.h x pila.c x pila.h x pila.c x pila.h x × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 14 pilaLlena(const tPila *p, unsigned cantBytes) int pilaLlena(const tPila *p, unsigned cantBytes) 15 16 return p->tope < cantBytes + sizeof(unsigned);</pre> tNodo *aux = (tNodo *)malloc(sizeof(tNodo)); 17 void *info = malloc(cantBytes); free (aux): < 20 free (info); 21 return aux == NULL || info == NULL; 22 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

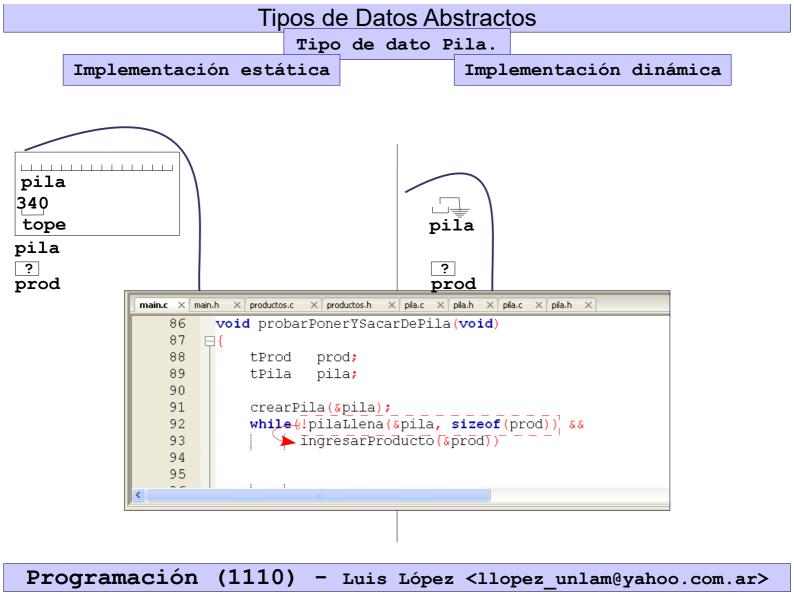
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pilaLlena pilaLlena pila 340 info tam sig tope pila pila 57 57 ? ? cantBytes cantBytes prod prod main.c × main.h X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X 86 void probarPonerYSacarDePila(void) 87 88 tProd prod; tPila. pila; × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × pilaLlena(const tPila *p, unsigned cantBytes) 14 int pilaLlena(const tPila *p, unsigned cantBytes) 115. 16 return p->tope < cantBytes + sizeof(unsigned);</pre> tNodo *aux = (tNodo *)malloc(sizeof(tNodo)); 17 void *info = malloc(cantBytes); free (aux): < 20 free (info); 21 return aux == NULL || info == NULL; 22 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

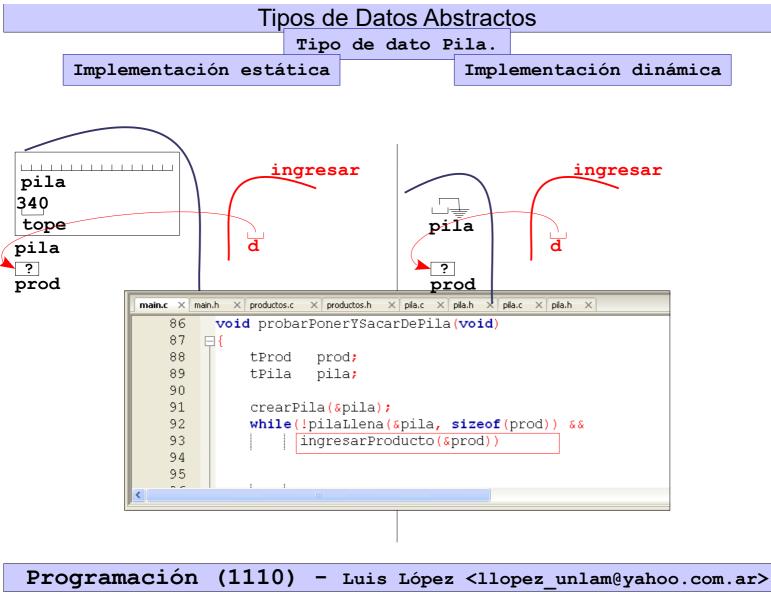
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pilaLlena pilaLlena pila 340 aux info tam sig tope pila info pila 57 57 ? ? cantBytes cantBytes prod prod X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X main.c X main.h 86 void probarPonerYSacarDePila(void) 87 88 tProd prod; t.Pila pila; × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 14 pilaLlena(const tPila *p, unsigned cantBytes) int pilaLlena(const tPila *p, unsigned cantBytes) 115 16 return p->tope < cantBytes + sizeof(unsigned);</pre> *aux = (tNodo *)malloc(sizeof(tNodo)); tNodo 17 void *info = malloc(cantBytes); free (aux): < 20 free (info); 21 return aux == NULL || info == NULL; 22 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

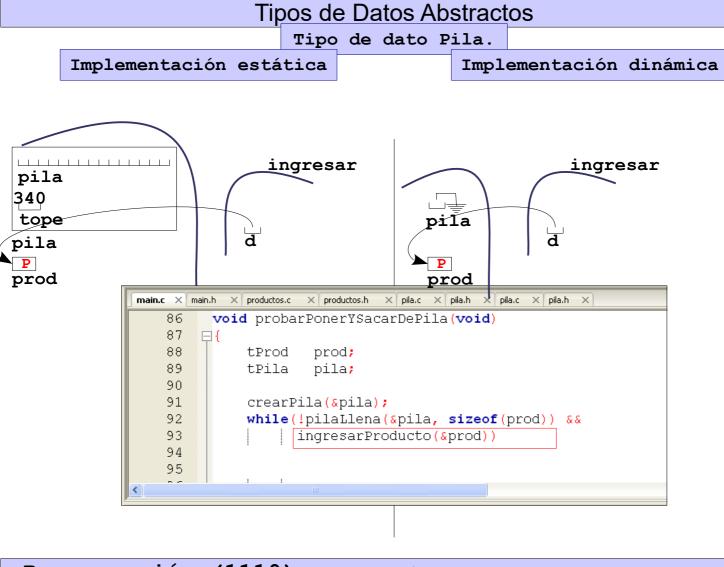
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pilaLlena pilaLlena pila 340 aux info tam sig tope pila info pila 57 57 ? ? cantBytes cantBytes prod prod X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X main.c X main.h 86 void probarPonerYSacarDePila(void) 87 88 tProd prod; t.Pila pila; × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 14 pilaLlena(const tPila *p, unsigned cantBytes) int pilaLlena(const tPila *p, unsigned cantBytes) 115 16 return p->tope < cantBytes + sizeof(unsigned);</pre> tNodo *aux = (tNodo *)malloc(sizeof(tNodo)); 17 void *info = malloc(cantBytes); free (aux); < 20 free(info); 21 return aux == NULL || info == NULL; 22 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

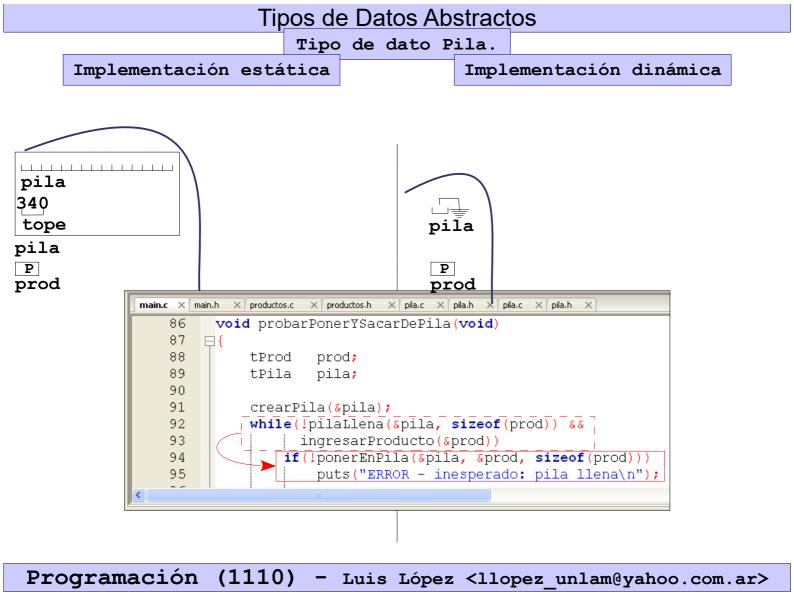
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pilaLlena pilaLlena pila 340 aux tope pila info pila 57 57 ? ? cantBytes cantBytes prod prod main.c × main.h X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X 86 void probarPonerYSacarDePila(void) 87 88 tProd prod; tPila. pila; × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × pilaLlena(const tPila *p, unsigned cantBytes) 14 int pilaLlena(const tPila *p, unsigned cantBytes) 115. 16 return p->tope < cantBytes + sizeof(unsigned);</pre> tNodo *aux = (tNodo *)malloc(sizeof(tNodo)); 17 void *info = malloc(cantBytes); free (aux); < 20 free (info); 21 return aux == NULL || info == NULL; 22 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pilaLlena pilaLlena pila 340 aux tope pila info pila 57 57 ? ? cantBytes cantBytes prod prod main.c × main.h X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X 86 void probarPonerYSacarDePila(void) 87 88 tProd prod; tPila. pila; × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 14 int pilaLlena(const tPila *p, unsigned cantBytes) int pilaLlena(const tPila *p, unsigned cantBytes) 15 return p->tope < cantBytes + sizeof(unsigned);</pre> 16 tNodo *aux = (tNodo *)malloc(sizeof(tNodo)); 17 void *info = malloc(cantBytes); free (aux): < 20 free (info); 21 return aux == NULL || info == NULL; 22 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>









Tipo de dato Pila. Implementación estática Implementación dinámica ponerEnPila ponerEnPila pila 340 tope pila pila Р cantBytes cantBytes prod prod × productos.c × productos.h × pila.c × pila.h main.c X main.h x pila.c x pila.h x 86 void probarPonerYSacarDePila(void) 87 88 tProd prod; 89 tPila pila; 90 91 crearPila(&pila); 92 while(!pilaLlena(&pila, sizeof(prod)) && 93 ingresarProducto(&prod)) if(!ponerEnPila(&pila, &prod, sizeof(prod))) 94 95 puts("ERROR - inesperado: pila llena\n");

- Luis López Lopez unlam@yahoo.com.ar>

Programación (1110)

Tipos de Datos Abstractos

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica ponerEnPila ponerEnPila pila 340 tope pila pila Р cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1: Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica ponerEnPila ponerEnPila pila 340 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if(p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1: Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica ponerEnPila ponerEnPila pila 283 tope pila pila cantBytes cantBytes prod prod main.c X main.h X productos.c X productos.h X pila.c X pila.h x pila.c x pila.h x void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) \square { if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0;) && p->tope -= cantBytes; memcpy(p->pila + p->tope, d, cantBytes); .zeof (prod))) p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1: Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica ponerEnPila ponerEnPila pila 283 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; memcpy(p->pila + p->tope, d, cantBytes); .zeof (prod))) p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica ponerEnPila ponerEnPila pila 279 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica ponerEnPila ponerEnPila pila 279 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                               ponerEnPila
                            ponerEnPila
pila
279
tope
                                               pila
pila
prod
                          cantBytes
                                               prod
                                                             cantBytes
                                  ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
                             int
              main.c × main.h
                 86
                       voi
                                 tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
     ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                     free (nue);
         return 0:
                                     return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                 memcpy(nue->info, d, cantBytes);
      p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                 nue->siq = *p;
      return 1:
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                    Luis López < llopez unlam@yahoo.com.ar>
```

```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                                          info tam sig
                                                              ponerEnPila
                            ponerEnPila
pila
279
                                                                 nue
tope
                                               pila
pila
prod
                          cantBytes
                                               prod
                                                             cantBytes
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                       voi
                                 tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL | |
  int
     ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                 memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                 nue->siq = *p;
     return 1:
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                    Luis López < llopez unlam@yahoo.com.ar>
```

```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                                         info tam sig
                                                              ponerEnPila
                            ponerEnPila
pila
279
                                                                 nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL | |
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if (p->tope < cantByte
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1:
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                    Luis López < llopez unlam@yahoo.com.ar>
```

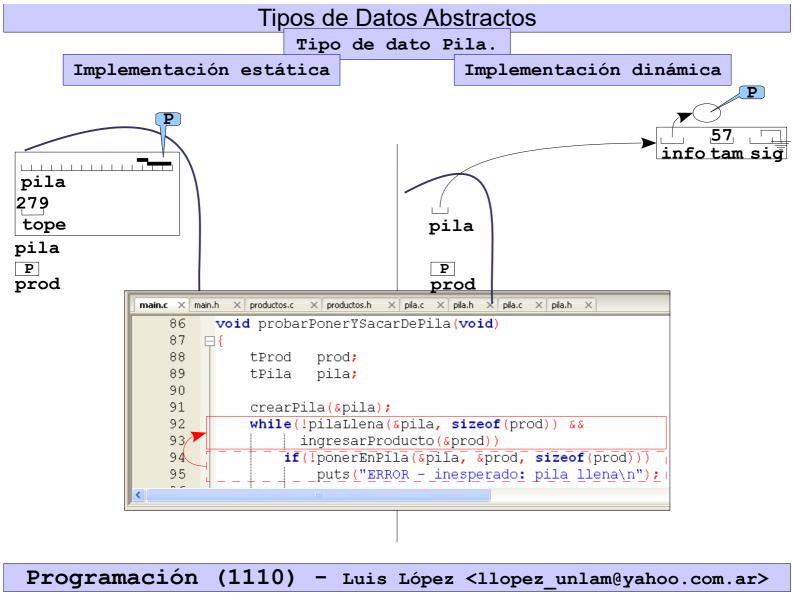
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                                          info tam sig
                                                              ponerEnPila
                            ponerEnPila
pila
279
                                                                 nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                       voi
                                 tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                 memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                 nue->siq = *p;
     return 1:
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                    Luis López < llopez unlam@yahoo.com.ar>
```

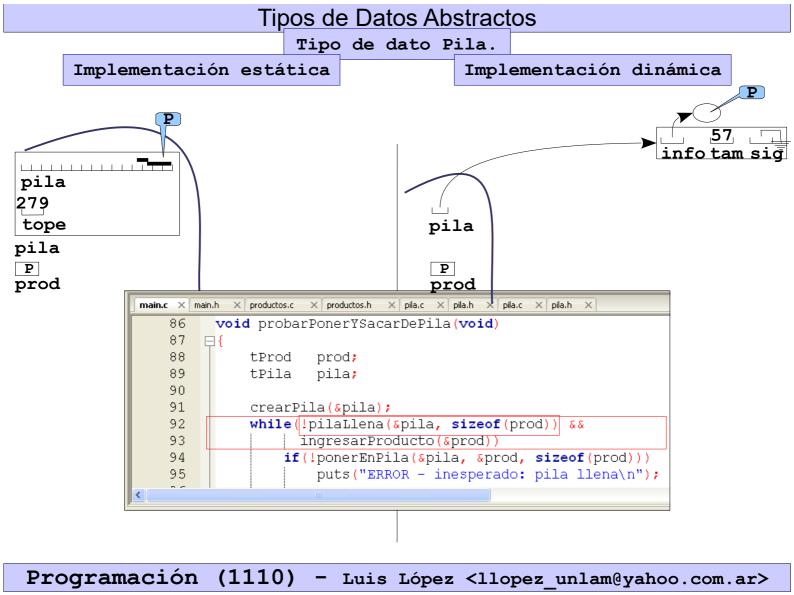
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                                         info tam sig
                                                              ponerEnPila
                           ponerEnPila
pila
279
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if (p->tope < cantByte
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1:
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                   Luis López < llopez unlam@yahoo.com.ar>
```

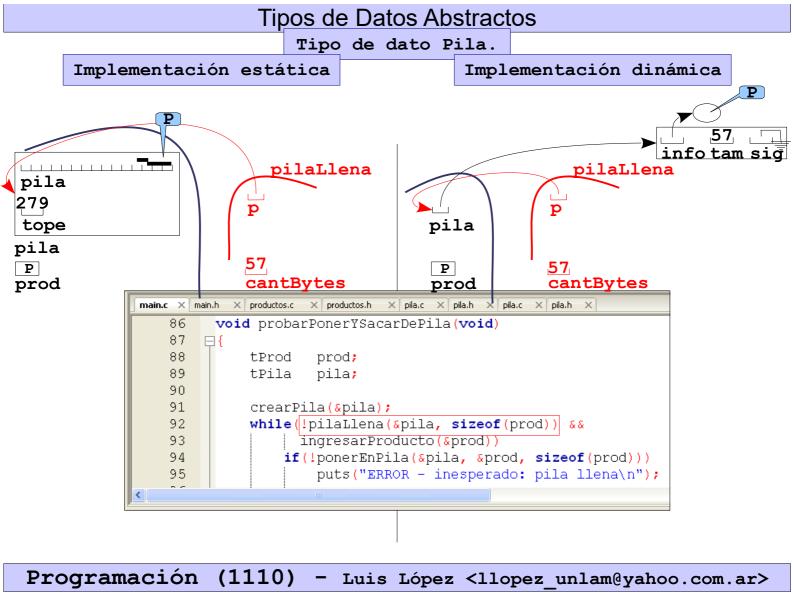
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                                               57
                                                                         info tam siq
                                                              ponerEnPila
                           ponerEnPila
pila
279
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if (p->tope < cantByte
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1:
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                   Luis López < llopez unlam@yahoo.com.ar>
```

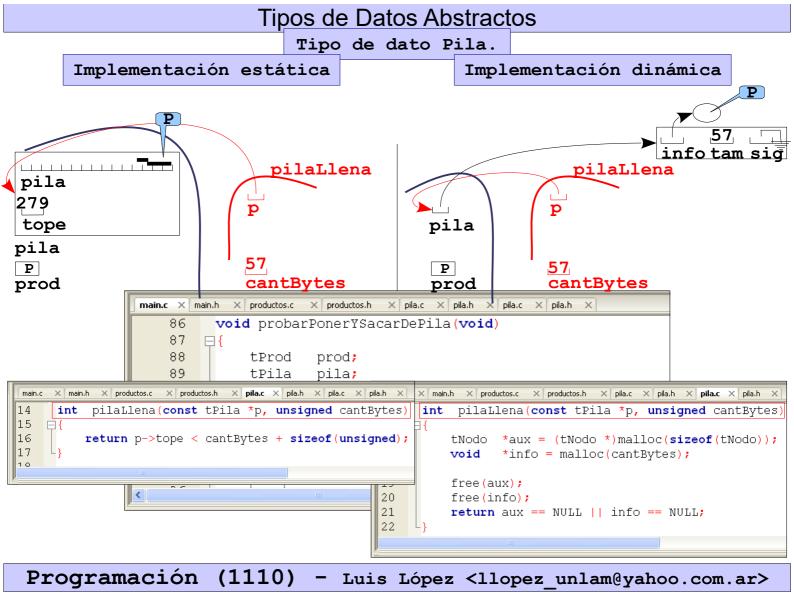
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                                               57
                                                                         info tam siq
                                                              ponerEnPila
                           ponerEnPila
pila
279
                                                                 nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL | |
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if (p->tope < cantByte
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                 nue->siq = *p;
     return 1:
                                 *p = nue;
                                 return 1:
 Programación (1110)
                                  - Luis López <llopez unlam@yahoo.com.ar>
```

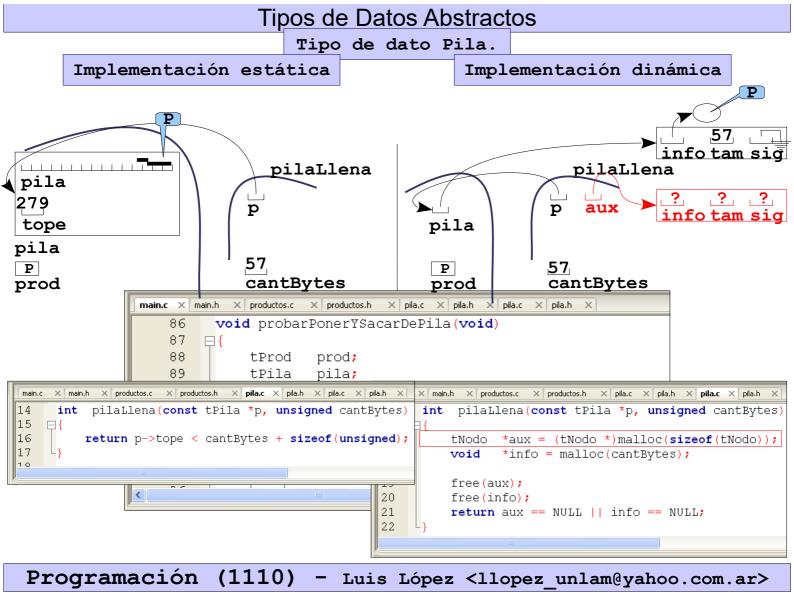
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                                               57
                                                                         info tam siq
                                                              ponerEnPila
                            ponerEnPila
pila
279
                                                                 nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL | |
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if (p->tope < cantByte
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
      return 1;
                                 *p = nue;
                                 return 1:
 Programación (1110)
                                  - Luis López <llopez unlam@yahoo.com.ar>
```

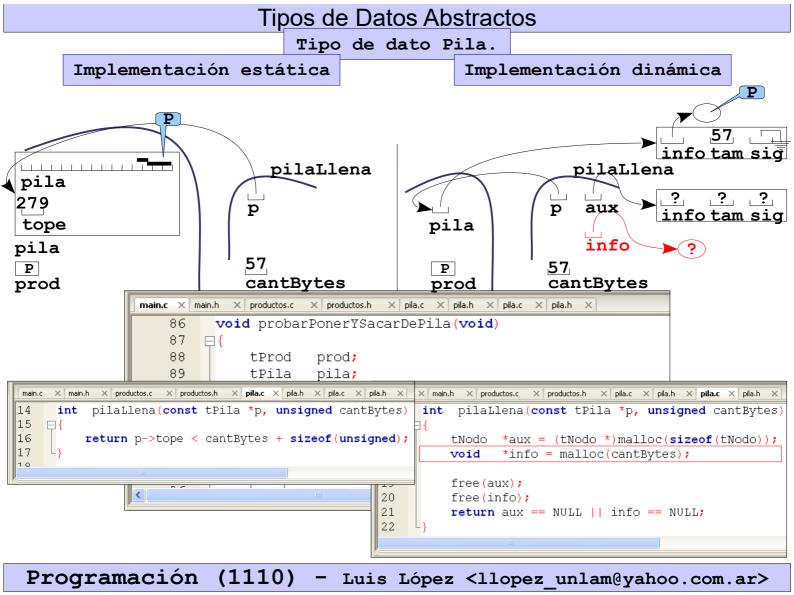


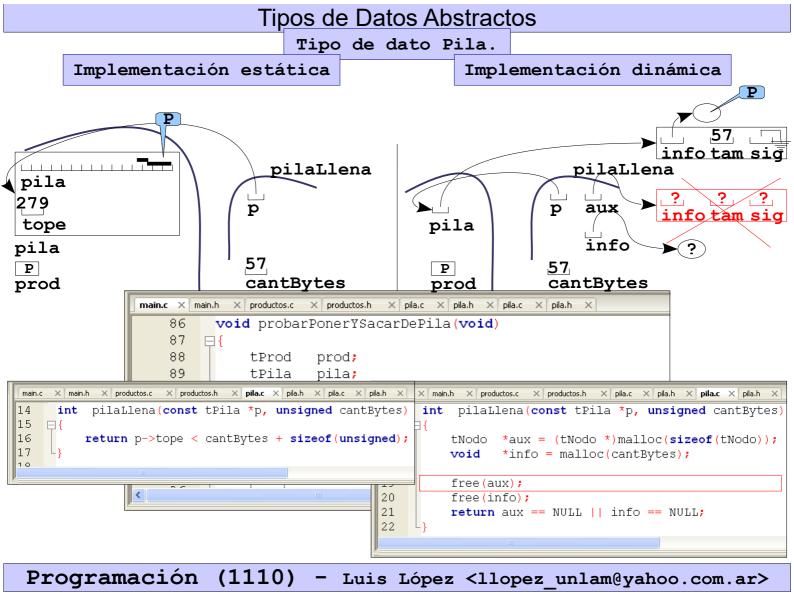


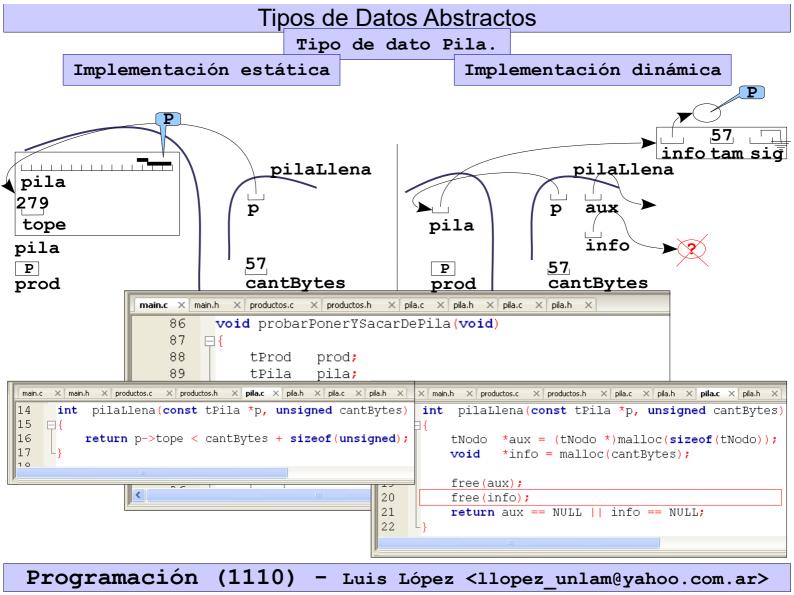


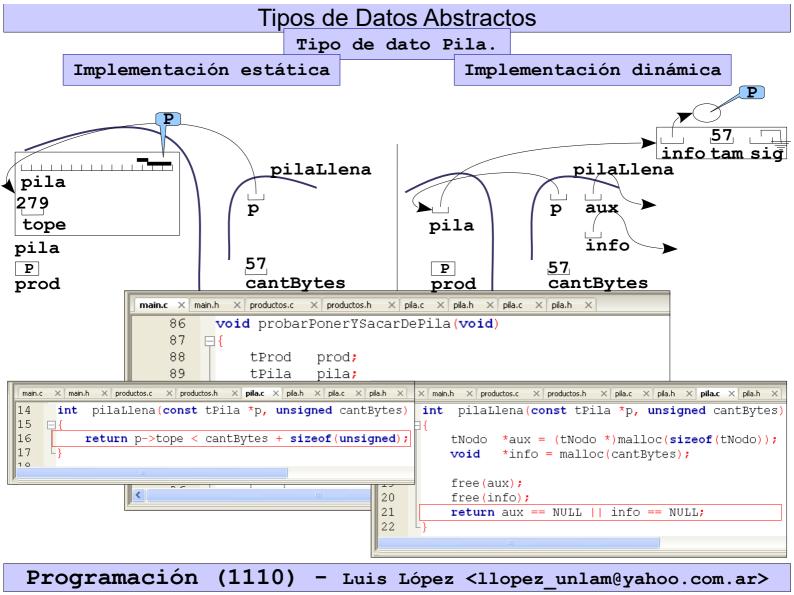


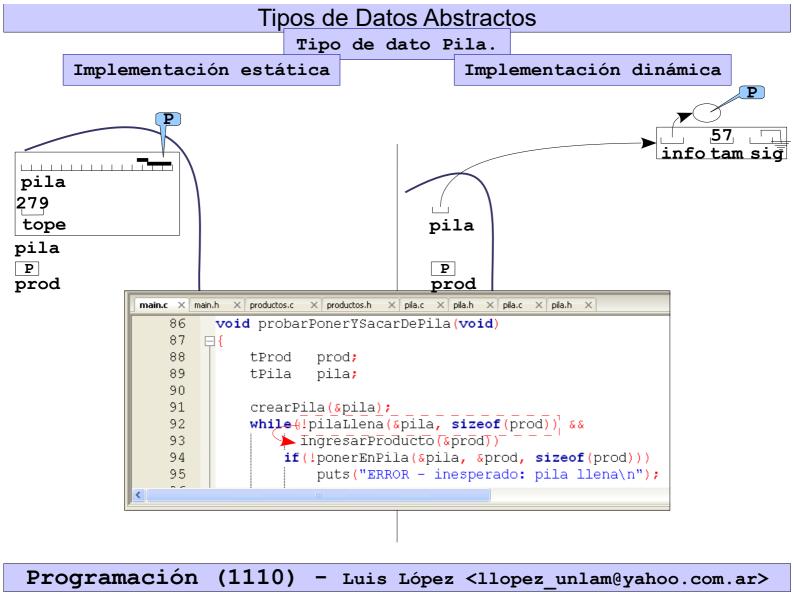


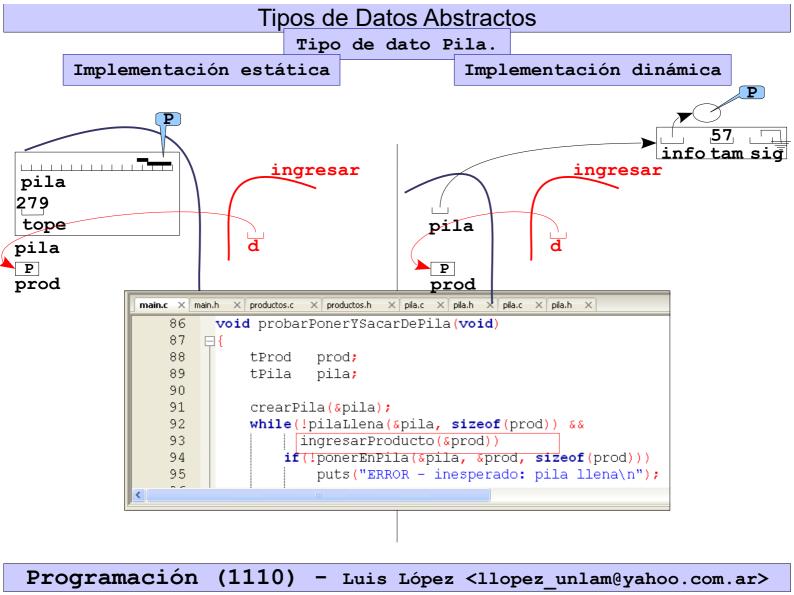


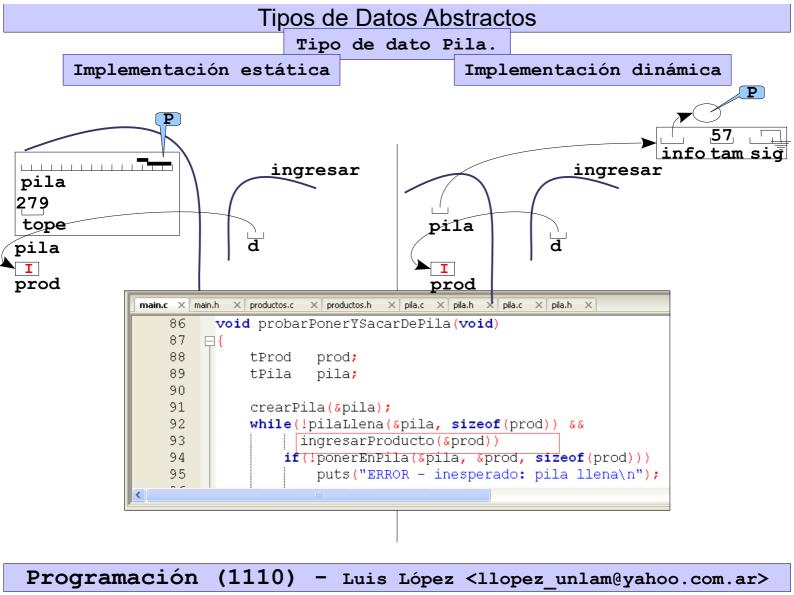


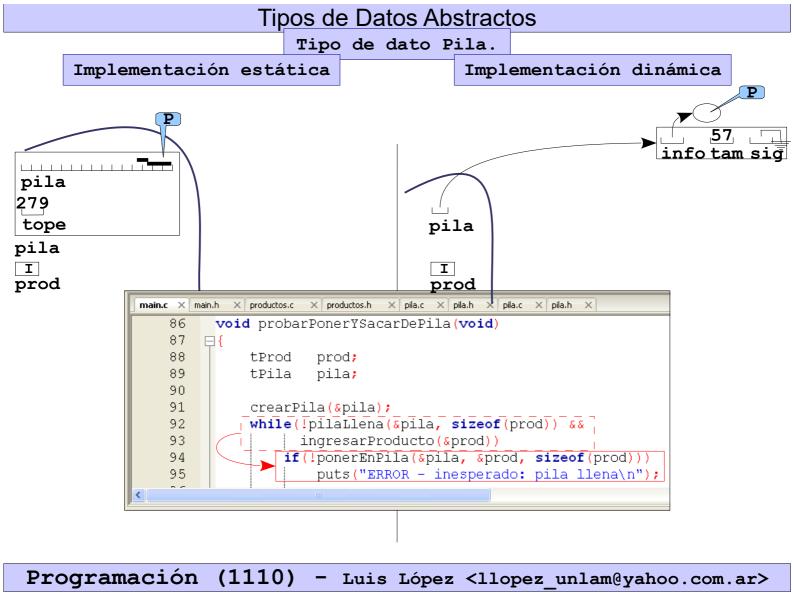


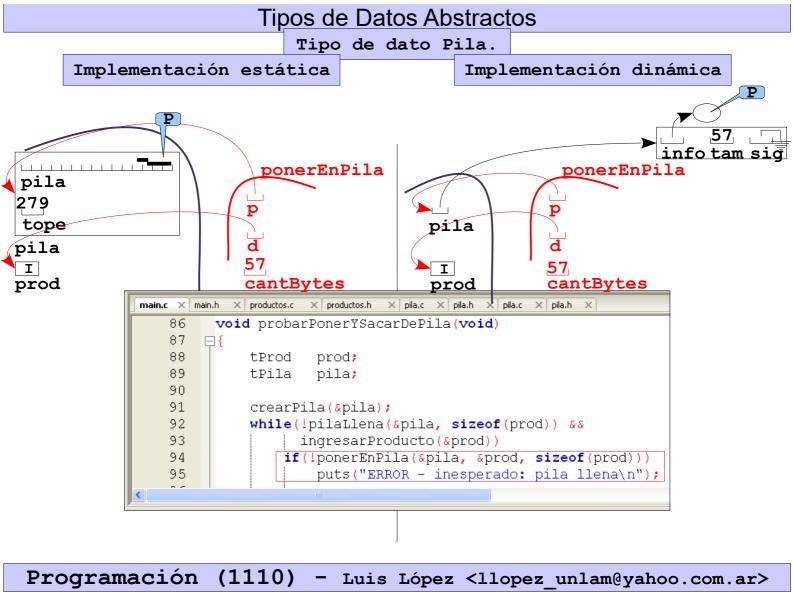












Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica info tam sia ponerEnPila ponerEnPila pila 279 tope pila pila I prod cantBytes cantBytes prod main.c X main.h X productos.c X productos.h X pila.c X pila.h x pila.c x pila.h x void probarPonerYSacarDePila(void) 86 × productos,c × productos,h × pila,c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica info tam sia ponerEnPila ponerEnPila pila 279 tope pila pila cantBytes cantBytes prod prod main.c X main.h X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if(p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica info tam sia ponerEnPila ponerEnPila pila tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos.c × productos.h × pila.c × pila.h x pila.c x pila.h x void probarPonerYSacarDePila(void) 86 main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) \square { if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0;) && p->tope -= cantBytes; memcpy(p->pila + p->tope, d, cantBytes); .zeof (prod))) p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica info tam sia ponerEnPila ponerEnPila pila 222 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; memcpy(p->pila + p->tope, d, cantBytes); .zeof (prod))) p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica info tam siq ponerEnPila ponerEnPila pila 218 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica info tam sia ponerEnPila ponerEnPila pila 218 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                                          info tam sia
                            ponerEnPila
                                                               ponerEnPila
pila
218
tope
                                               pila
pila
prod
                          cantBytes
                                               prod
                                                             cantBytes
                                  ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
                             int
              main.c × main.h
                 86
                       voi
                                 tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                     free (nue);
         return 0:
                                     return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                 memcpy(nue->info, d, cantBytes);
      p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                 nue->siq = *p;
      return 1;
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                    Luis López < llopez unlam@yahoo.com.ar>
```

```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                       info tam siq
                                                                        info tam siq
                           ponerEnPila
                                                              ponerEnPila
pila
218
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                               prod
                                                            cantBytes
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL | |
  int
     ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if (p->tope < cantByte
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1:
                                 *p = nue;
                                return 1;
 Programación (1110)
                                   Luis López < llopez unlam@yahoo.com.ar>
```

```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                       info tam sig
                                                                        info tam siq
                                                              ponerEnPila
                           ponerEnPila
pila
218
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                               prod
                                                            cantBytes
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL | |
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if (p->tope < cantByte
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1:
                                 *p = nue;
                                return 1;
 Programación (1110)
                                   Luis López < llopez unlam@yahoo.com.ar>
```

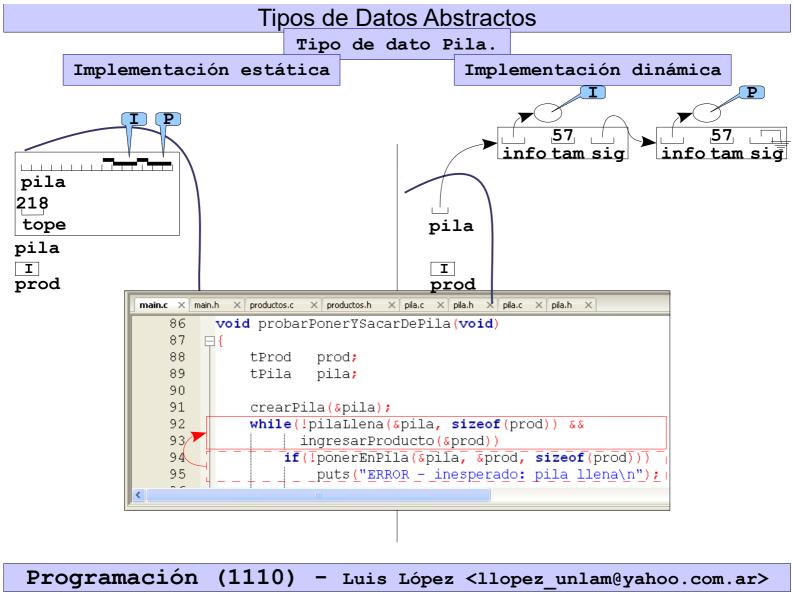
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                       info tam sig
                                                                         info tam siq
                                                              ponerEnPila
                            ponerEnPila
pila
218
                                                                 nue
tope
                                               pila
pila
prod
                          cantBytes
                                               prod
                                                            cantBytes
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1;
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                    Luis López < llopez unlam@yahoo.com.ar>
```

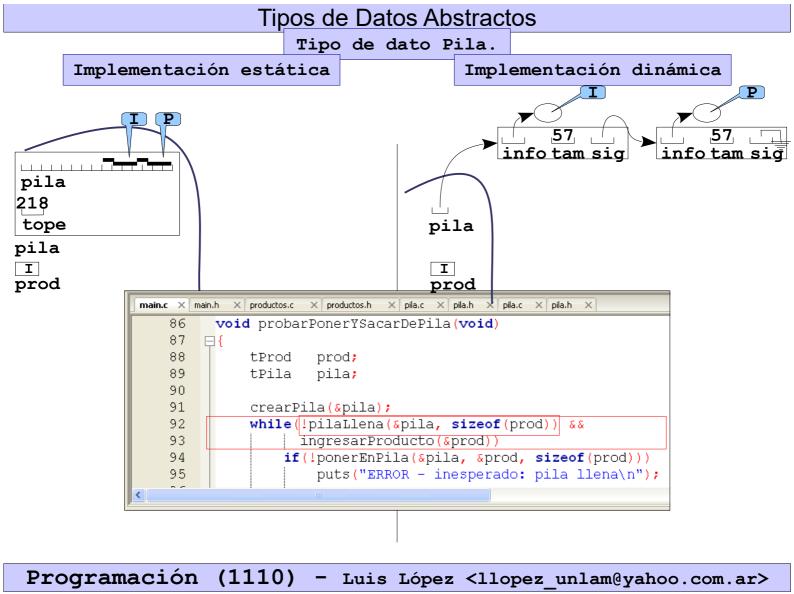
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                       info tam sig
                                                                        info tam siq
                           ponerEnPila
                                                              ponerEnPila
pila
218
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                               prod
                                                            cantBytes
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1;
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                   Luis López < llopez unlam@yahoo.com.ar>
```

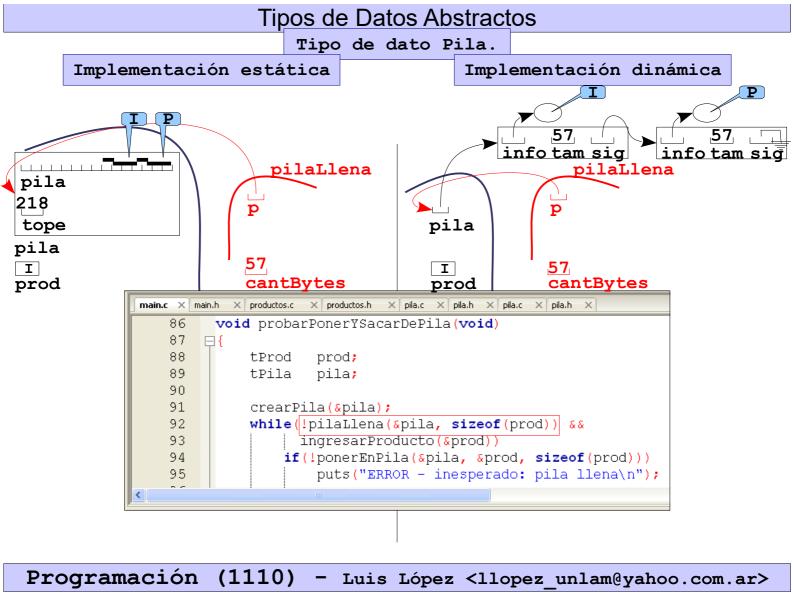
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                             57
                                                       info tam sig
                                                                         info tam sia
                                                              ponerEnPila
                            ponerEnPila
pila
218
                                                                 nue
tope
                                               pila
pila
prod
                          cantBytes
                                               prod
                                                            cantBytes
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                 tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                 memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                 nue->siq = *p;
     return 1:
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                    Luis López < llopez unlam@yahoo.com.ar>
```

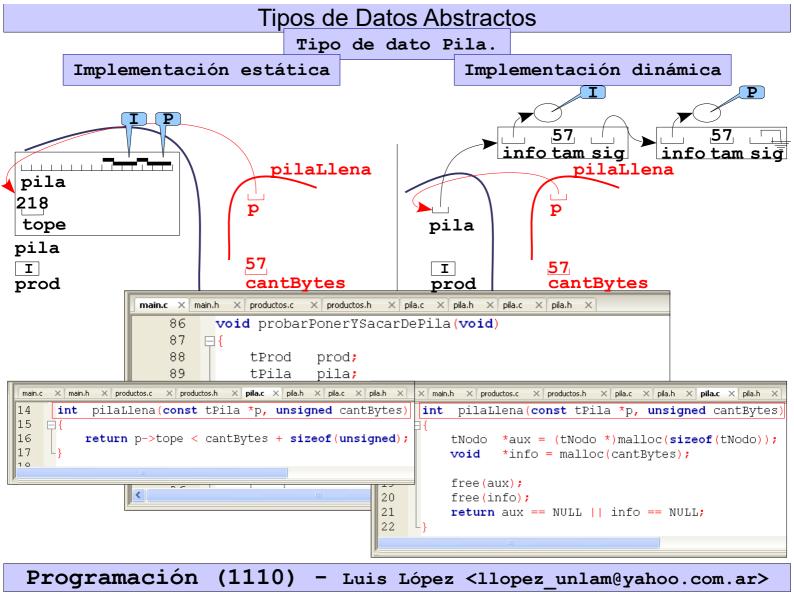
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                             57
                                                       info tam sig
                                                                         info tam siq
                            ponerEnPila
                                                              ponerEnPila
pila
218
                                                                 nue
tope
                                               pila
pila
prod
                          cantBytes
                                               prod
                                                            cantBytes
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                 nue->siq = *p;
     return 1:
                                 *p = nue;
                                 return 1:
 Programación (1110)
                                  - Luis López <llopez unlam@yahoo.com.ar>
```

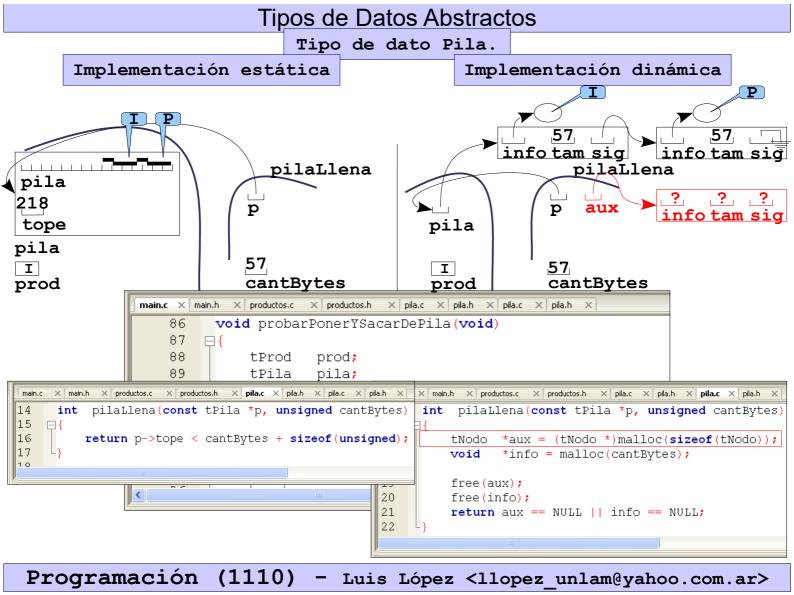
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam siq ponerEnPila ponerEnPila pila 218 nue tope pila pila prod cantBytes prod cantBytes int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) main.c × main.h 86 voi tNodo *nue; main.h × productos.c × product if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL || int ponerEnPila(tPila *r (nue->info = malloc(cantBytes)) == NULL) \square { if (p->tope < cantByte free (nue); return 0: return 0: p->tope -= cantBytes; memcpy(p->pila + p->t memcpy(nue->info, d, cantBytes); p->tope -= sizeof(uns nue->tamInfo = cantBytes; memcpy(p->pila + p->t nue->siq = *p; return 1; *p = nue;return 1: Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

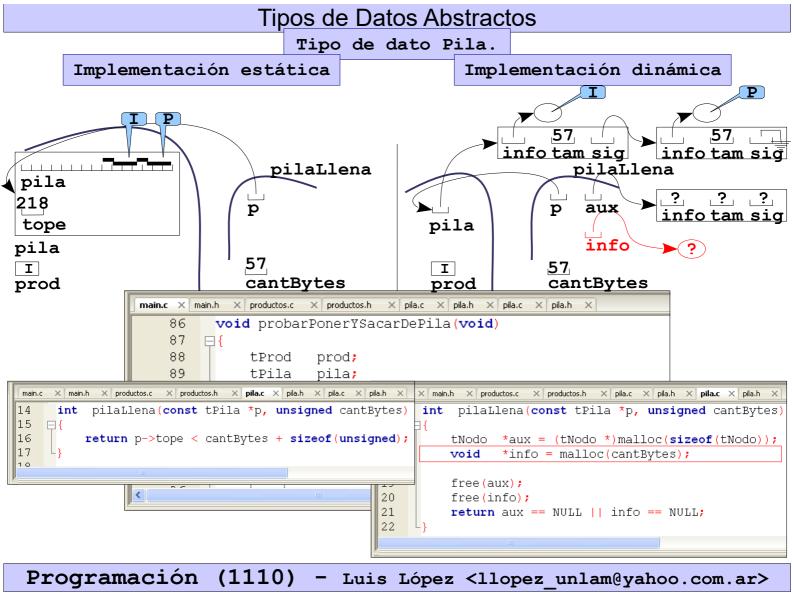


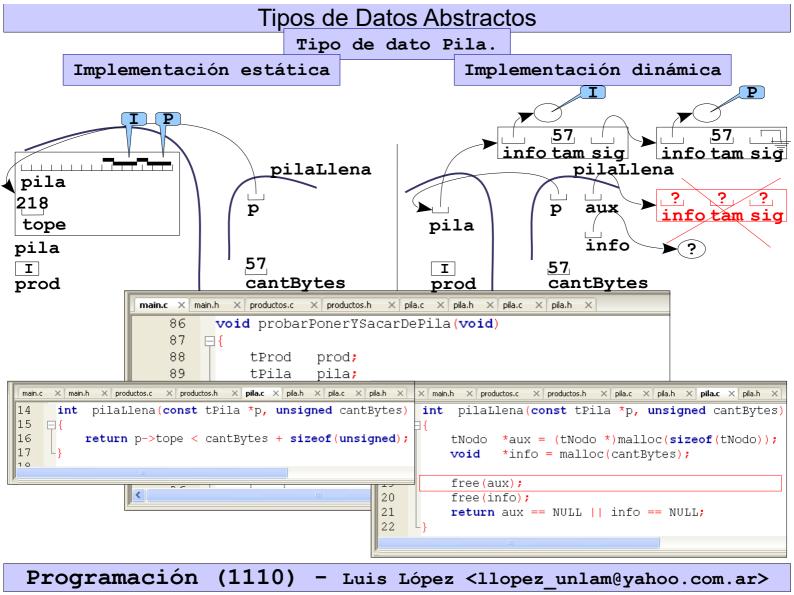


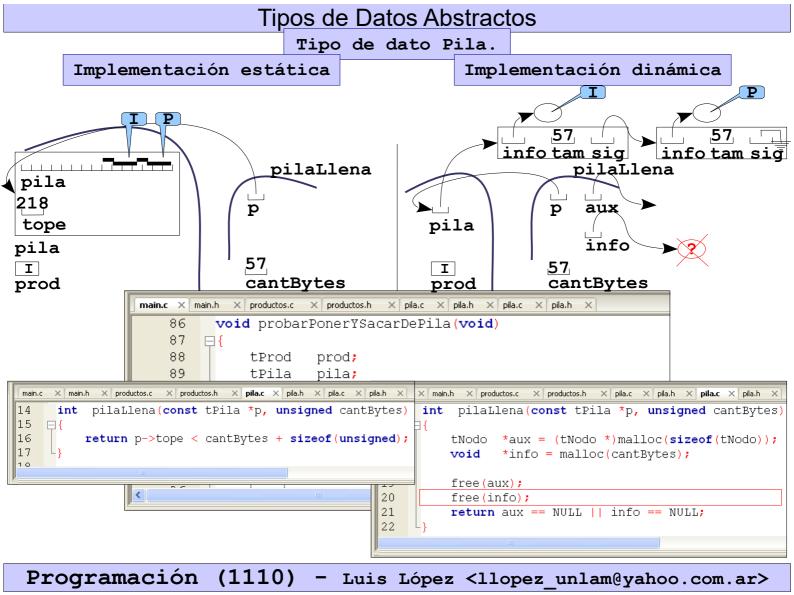


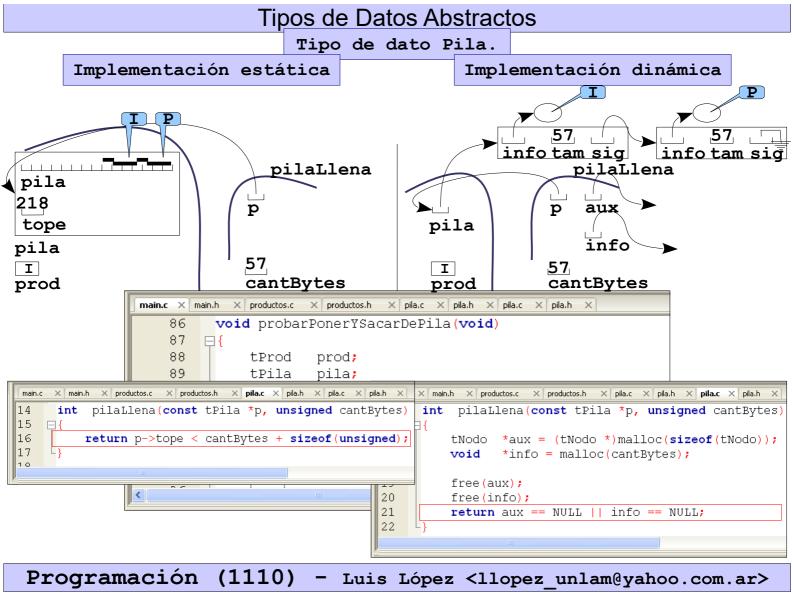


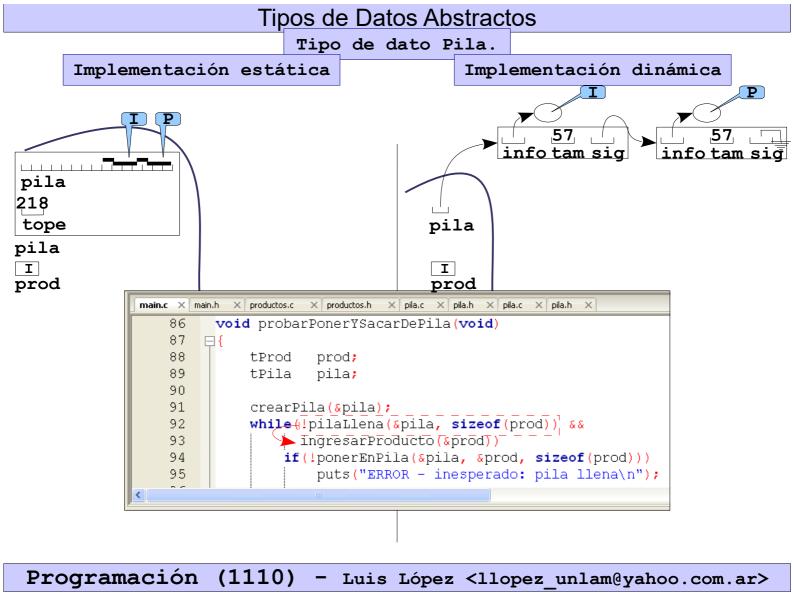


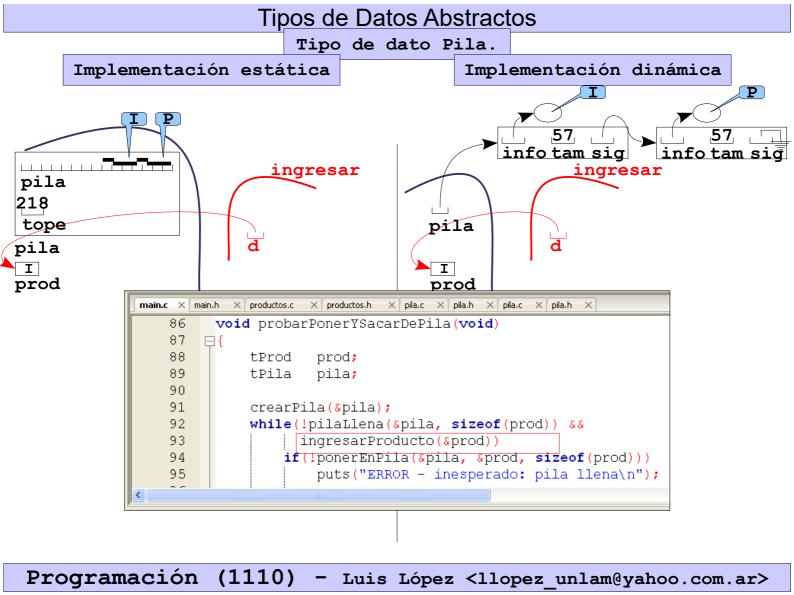


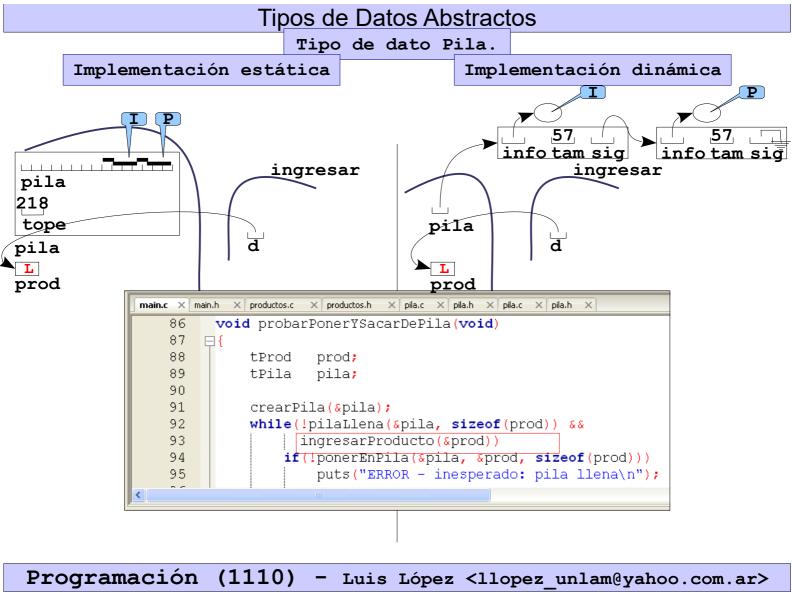


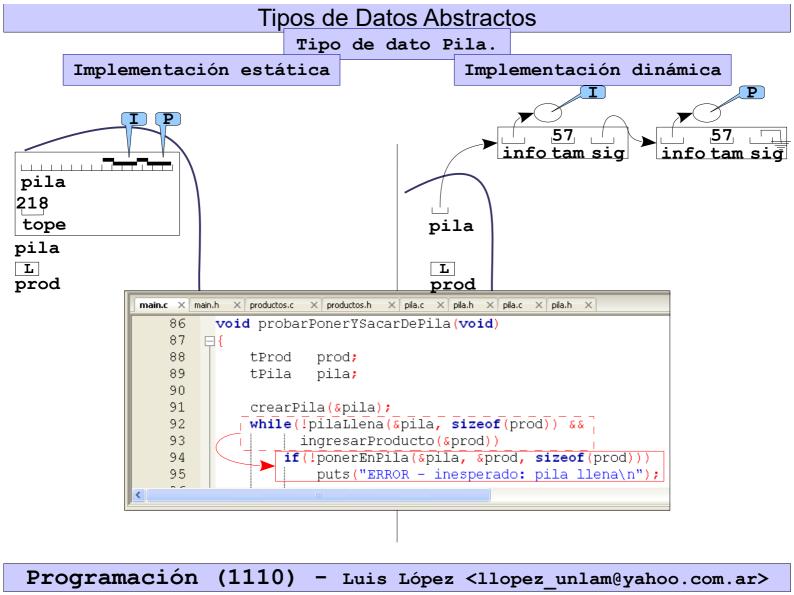


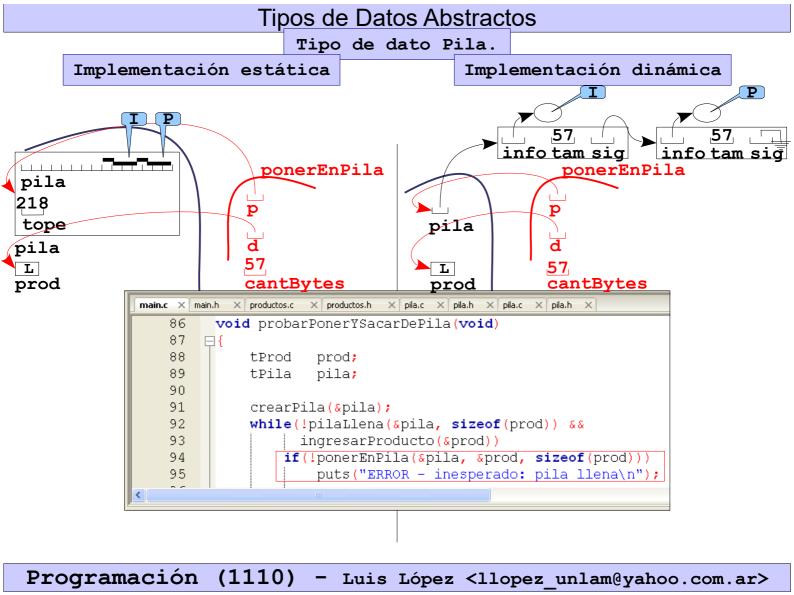












Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam siq ponerEnPila ponerEnPila pila 218 tope pila pila L cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 × productos,c × productos,h × pila,c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam siq ponerEnPila ponerEnPila pila 218 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if(p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam siq ponerEnPila ponerEnPila pila 161 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) \square { if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0;) && p->tope -= cantBytes; memcpy(p->pila + p->tope, d, cantBytes); .zeof (prod))) p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam siq ponerEnPila ponerEnPila pila 161 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; memcpy(p->pila + p->tope, d, cantBytes); .zeof (prod))) p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam siq ponerEnPila ponerEnPila pila 157 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam siq ponerEnPila ponerEnPila pila 157 tope pila pila cantBytes cantBytes prod prod main.c × main.h × productos,c × productos,h × pila.c × pila.h × pila.c × pila.h × void probarPonerYSacarDePila(void) 86 main,h × productos,c × productos,h × pila.c × pila,h × pila,c × pila,h × int ponerEnPila(tPila *p, const void *d, unsigned cantBytes) if (p->tope < cantBytes + sizeof(unsigned))</pre> return 0:) && p->tope -= cantBytes; .zeof (prod))) memcpy(p->pila + p->tope, d, cantBytes); p->tope -= sizeof(unsigned); .la llena\n"); memcpy(p->pila + p->tope, &cantBytes, sizeof(unsigned)); return 1; Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación dinámica Implementación estática 57 info tam sig info tam siq ponerEnPila ponerEnPila pila 157 tope pila pila cantBytes prod prod cantBytes ponerEnPila(tPila *p, const void *d, unsigned cantBytes) int main.c × main.h 86 voi tNodo *nue; main.h × productos.c × product if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL || int ponerEnPila(tPila *r (nue->info = malloc(cantBytes)) == NULL) \square { if(p->tope < cantByte</pre> free (nue); return 0: return 0: p->tope -= cantBytes; memcpy(p->pila + p->t memcpy(nue->info, d, cantBytes); p->tope -= sizeof(uns nue->tamInfo = cantBytes; memcpy(p->pila + p->t nue->siq = *p; return 1; *p = nue;return 1; Programación (1110) Luis López < llopez unlam@yahoo.com.ar>

```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
                                                   Implementación dinámica
      Implementación estática
                                                            57
                                    info tam sig
                                                      info tam sig
                                                                        info tam sig
                           ponerEnPila
                                                              ponerEnPila
pila
157
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
       main.h × productos.c × product
                                if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL | |
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if (p->tope < cantByte
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1:
                                *p = nue;
                                return 1;
 Programación (1110)
                                   Luis López < llopez unlam@yahoo.com.ar>
```

```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                            57
                                    info tam sig
                                                      info tam sig
                                                                        info tam sia
                           ponerEnPila
                                                              ponerEnPila
pila
157
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
       main.h × productos.c × product
                                if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL | |
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if (p->tope < cantByte
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1;
                                *p = nue;
                                return 1;
 Programación (1110)
                                   Luis López < llopez unlam@yahoo.com.ar>
```

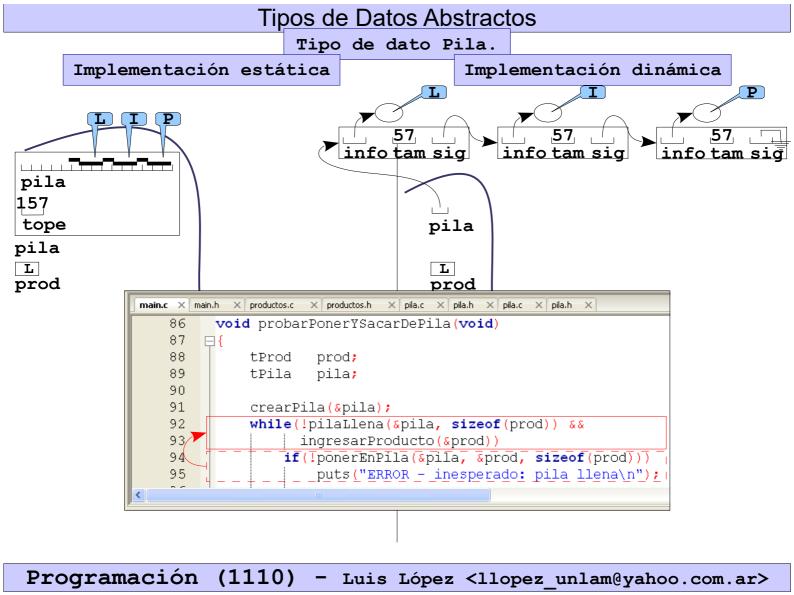
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                             57
                                   info tam sig
                                                      info tam sig
                                                                         info tam sia
                           ponerEnPila
                                                              ponerEnPila
pila
157
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1;
                                 *p = nue;
                                return 1;
 Programación (1110)
                                    Luis López < llopez unlam@yahoo.com.ar>
```

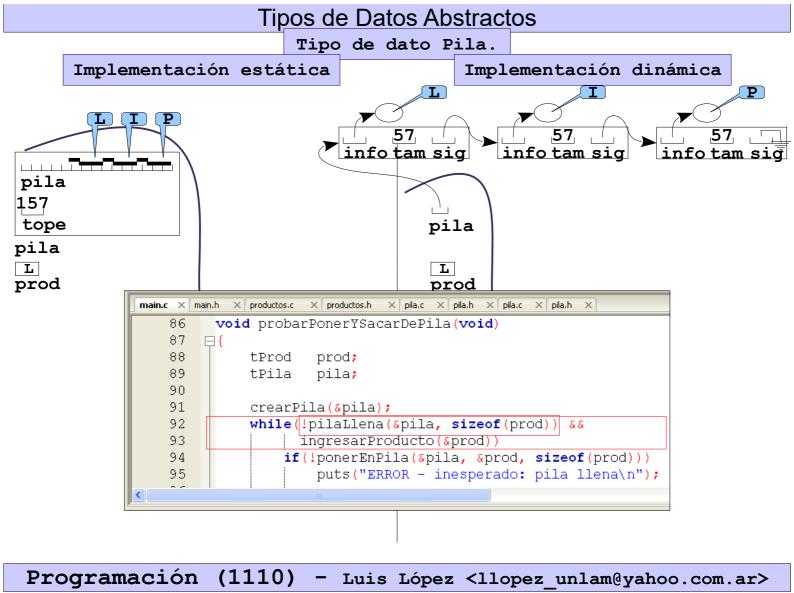
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                             57
                                    info tam sig
                                                      info tam sig
                                                                        info tam sig
                           ponerEnPila
                                                              ponerEnPila
pila
157
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
      main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1;
                                 *p = nue;
                                return 1;
 Programación (1110)
                                   Luis López < llopez unlam@yahoo.com.ar>
```

```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                             57
                                    info tam sig
                                                       info tam sig
                                                                         info tam siq
                           ponerEnPila
                                                              ponerEnPila
pila
157
                                                                 nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                 nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1:
                                 *p = nue;
                                 return 1;
 Programación (1110)
                                   Luis López < llopez unlam@yahoo.com.ar>
```

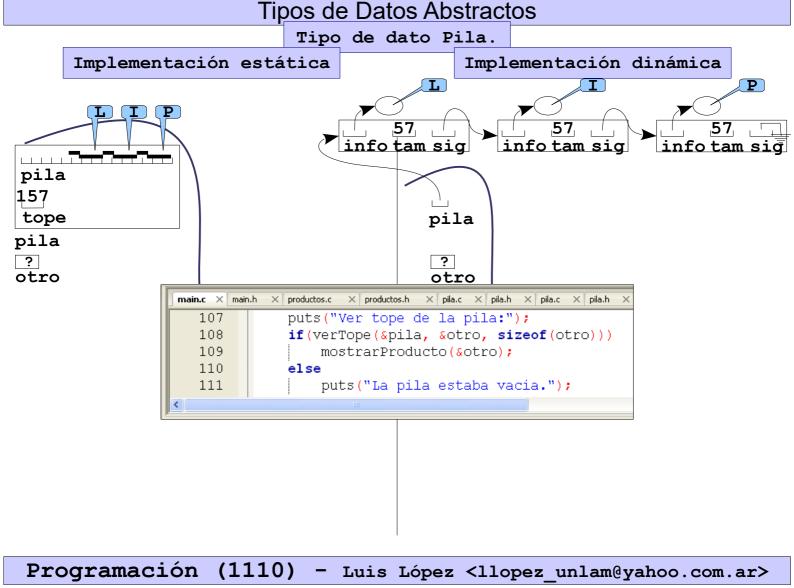
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                             57
                                    info tam sig
                                                      info tam sig
                                                                        info tam sia
                           ponerEnPila
                                                              ponerEnPila
pila
157
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1:
                                 *p = nue;
                                return 1:
 Programación (1110)
                                  - Luis López <llopez unlam@yahoo.com.ar>
```

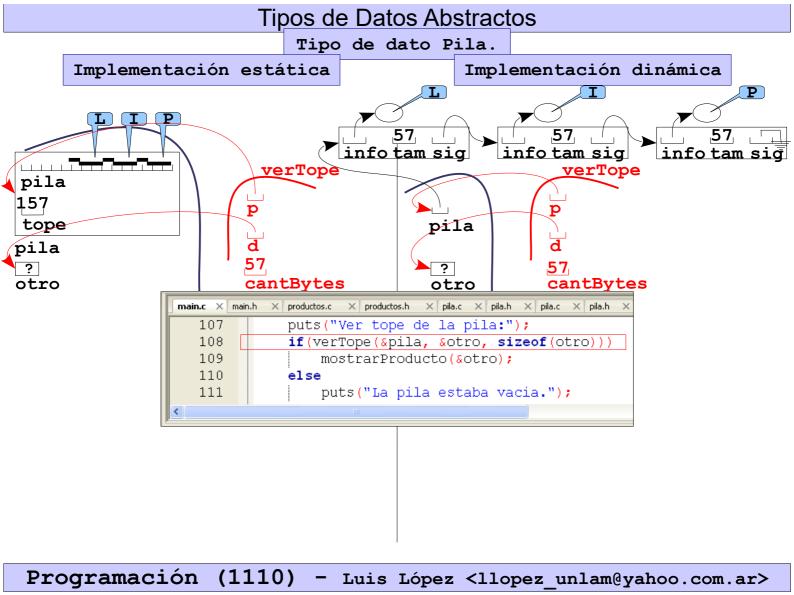
```
Tipos de Datos Abstractos
                                Tipo de dato Pila.
      Implementación estática
                                                   Implementación dinámica
                                                             57
                                    info tam sig
                                                      info tam sig
                                                                         info tam sia
                           ponerEnPila
                                                              ponerEnPila
pila
157
                          p
                                                                nue
tope
                                               pila
pila
prod
                          cantBytes
                                                            cantBytes
                                               prod
                             int ponerEnPila(tPila *p, const void *d, unsigned cantBytes)
              main.c × main.h
                 86
                      voi
                                tNodo *nue;
       main.h × productos.c × product
                                 if((nue = (tNodo *)malloc(sizeof(tNodo))) == NULL ||
  int
      ponerEnPila(tPila *r
                                    (nue->info = malloc(cantBytes)) == NULL)
\square{
      if(p->tope < cantByte</pre>
                                    free (nue);
         return 0:
                                    return 0:
      p->tope -= cantBytes;
     memcpy(p->pila + p->t
                                memcpy(nue->info, d, cantBytes);
     p->tope -= sizeof(uns
                                nue->tamInfo = cantBytes;
     memcpy(p->pila + p->t
                                nue->siq = *p;
     return 1;
                                 *p = nue;
                                return 1:
 Programación (1110)
                                  - Luis López <llopez unlam@yahoo.com.ar>
```











Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 57 info tam sig info tam sig info tam siq verTope verTope pila 157 tope pila pila cantBytes cantBytes otro otro main.c × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 107 nuts("Ver tope de la nila:"): btro))) × main.h × productos.h X pila.c X pila.h X pila.c X pila.h X main.c × productos.c 30 int verTope(const tPila *p, void *d, unsigned cantBytes) 31 \square 32 unsigned tamInfo; 33 34 if(p->tope == TAM PILA) 35 return 0; 36 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 37 memcpy(d, p->pila + p->tope + sizeof(unsigned), 38 minimo(cantBytes, tamInfo)); 39 return 1; 40 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación dinámica Implementación estática 57 info tam sig info tam sig info tam siq verTope verTope pila 157 tamInfo tope pila pila cantBytes otro otro cantBytes main.c × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 107 nuts("Ver tope de la nila:"): btro))) × main.h X pila.c X pila.h X pila.c X pila.h main.c productos.c × productos.h verTope(const tPila *p, void *d, unsigned cantBytes) 30 31 □ { 32 unsigned tamInfo; 33 34 if(p->tope == TAM PILA) 35 return 0; 36 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 37 memcpy(d, p->pila + p->tope + sizeof(unsigned), 38 minimo(cantBytes, tamInfo)); 39 return 1; 40 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación dinámica Implementación estática 57 info tam sig info tam sig info tam siq verTope verTope pila 57 tamInfo 157 tope pila pila cantBytes cantBytes otro otro main.c × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 107 nuts("Ver tope de la nila:"): btro))) × main.h X pila.c X pila.h X pila.c X pila.h main.c productos.c × productos.h 30 verTope(const tPila *p, void *d, unsigned cantBytes) 31 ⊟{ 32 unsigned tamInfo; 33 34 if(p->tope == TAM PILA) 35 return 0; 36 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 37 memcpy(d, p->pila + p->tope + sizeof(unsigned), 38 minimo(cantBytes, tamInfo)); 39 return 1; 40 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

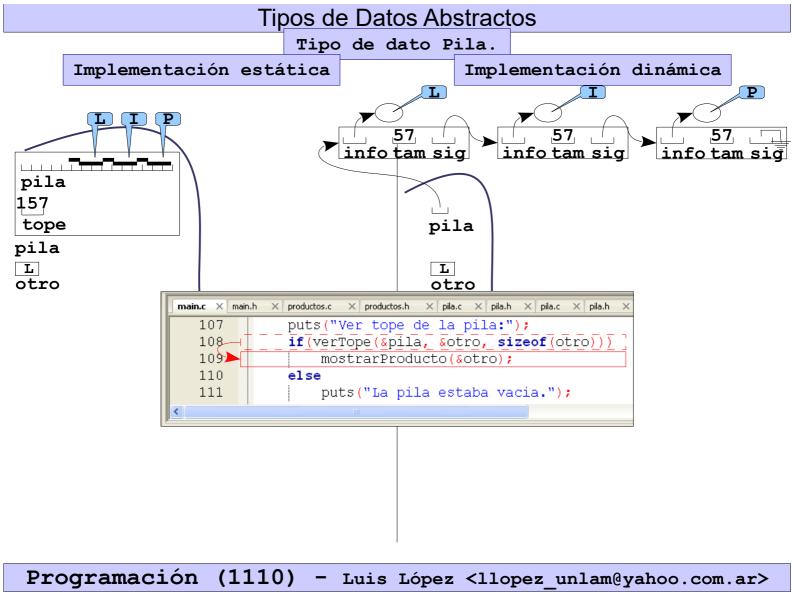
Tipos de Datos Abstractos Tipo de dato Pila. Implementación dinámica Implementación estática 57 57 info tam sig info tam sig info tam siq verTope verTope pila 57 157 tamInfo tope pila pila cantBytes cantBytes otro otro main.c × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 107 nuts("Ver tope de la pila:"): btro))) × main.h X pila.c X pila.h X pila.c X pila.h main.c productos.c × productos.h 30 verTope(const tPila *p, void *d, unsigned cantBytes) 31 ⊟{ 32 unsigned tamInfo; 33 34 if(p->tope == TAM PILA) 35 return 0; 36 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 37 memcpy(d, p->pila + p->tope + sizeof(unsigned), 38 minimo(cantBytes, tamInfo)); 39 return 1; 40 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación dinámica Implementación estática 57 info tam sig info tam sig info tam siq verTope verTope pila 157 tamInfo tope pila pila L cantBytes cantBytes otro otro main.c × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 107 nuts("Ver tone de la nila:"): × main.h × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × main.c productos.c × pro main.c 41 int verTope(const tPila *p, void *d, unsigned cantBytes) 30 int verTope(const 42 \square { 31 □ { 32 unsigned tamIn: 43 if(*p == NULL) return 0; 33 44 **if**(p->tope == 1 45 memcpy(d, (*p)->info, minimo(cantBytes, (*p)->tamInfo)); 34 35 return 0; | 46 return 1; memcpy(&tamInfd 47 36 37 memcpy(d, p->pills minimo(cantBytes, tamInfo)); 38 39 return 1; 40 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

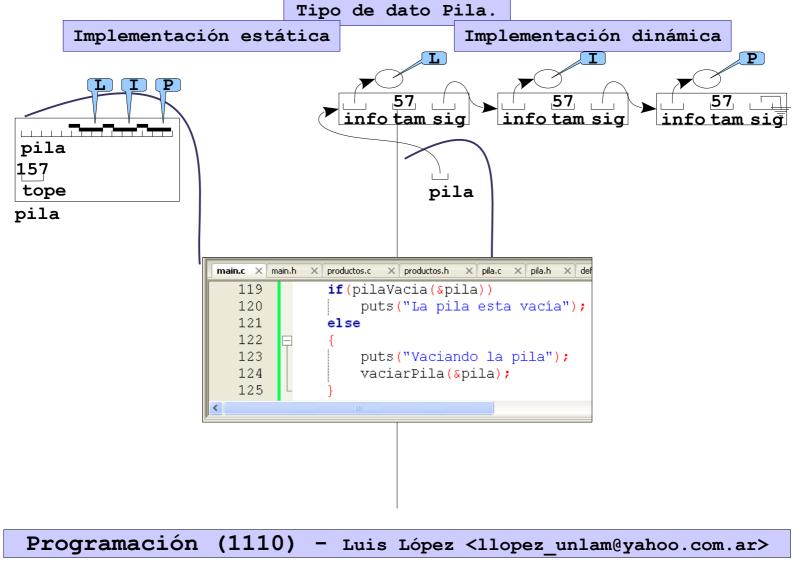
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam sig info tam siq verTope verTope pila 157 tamInfo tope pila pila 57 cantBytes cantBytes otro otro main.c × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 107 nuts("Ver tone de la nila:"): × main.h X productos.h X pila.c X pila.h X pila.c X pila.h X main.c productos.c × pro main.c X main.h X productos.c int verTope(const tPila *p, void *d, unsigned cantBytes) 30 int verTope(const 41 42 31 32 if(*p == NULL) unsigned tamIni 43 44 return 0; 33 **if**(p->tope == 1 45 memcpy(d, (*p)->info, minimo(cantBytes, (*p)->tamInfo)); 34 35 return 0; 11146 return 1; memcpy(&tamInfd 47 36 37 memcpy(d, p->pills minimo(cantBytes, tamInfo)); 38 39 return 1; 40 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación dinámica Implementación estática 57 info tam sig info tam sig info tam siq verTope verTope pila 157 tamInfo tope pila pila 57 L cantBytes cantBytes otro otro main.c × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 107 nuts("Ver tone de la nila:"): × main.h X productos.h X pila.c X pila.h X pila.c X pila.h X main.c productos.c × pro main.c X main.h X productos.c int verTope(const tPila *p, void *d, unsigned cantBytes) 30 int verTope(const 41 42 31 □{ \square 32 unsigned tamIn: 43 if(*p == NULL) 44 return 0; 33 memcpy(d, (*p)->info, minimo(cantBytes, (*p)->tamInfo)); **if**(p->tope == 1 45 34 35 return 0; 11146 return 1; memcpy(&tamInfd 47 36 37 memcpy(d, p->pi minimo(cantBytes, tamInfo)); 38 39 return 1; 40 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

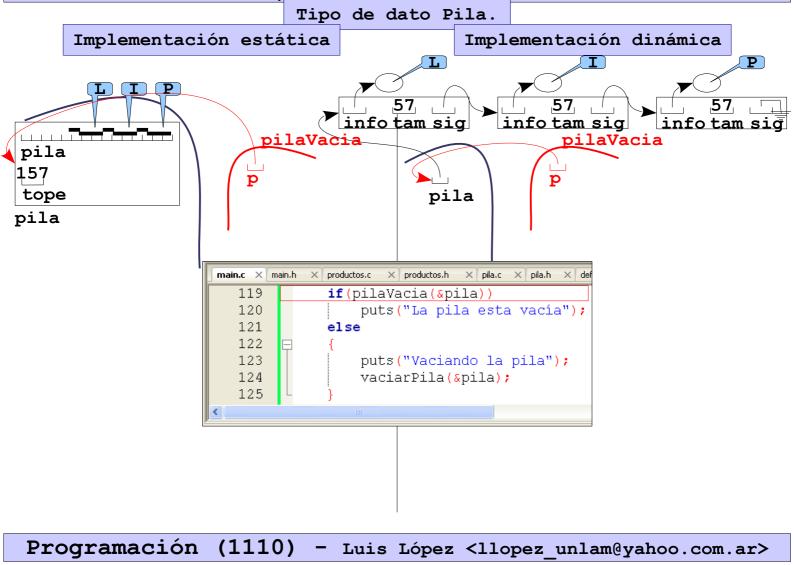
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam sig info tam siq verTope verTope pila 157 tamInfo tope pila pila 57 L cantBytes cantBytes otro otro main.c × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × 107 nuts("Ver tone de la nila:"): X productos.h X pila.c X pila.h X pila.c X pila.h X × main.h main.c productos.c × pro main.c X main.h X productos.c int verTope(const tPila *p, void *d, unsigned cantBytes) 30 int verTope(const 41 42 31 □{ \square 32 unsigned tamIn: 43 if(*p == NULL) return 0; 33 44 **if**(p->tope == 1 45 memcpy(d, (*p)->info, minimo(cantBytes, (*p)->tamInfo)); 34 35 return 0; III46 return 1; memcpy(&tamInfd 47 36 37 memcpy(d, p->pills 38 minimo(cantBytes, tamInfo)); 39 return 1; 40 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>



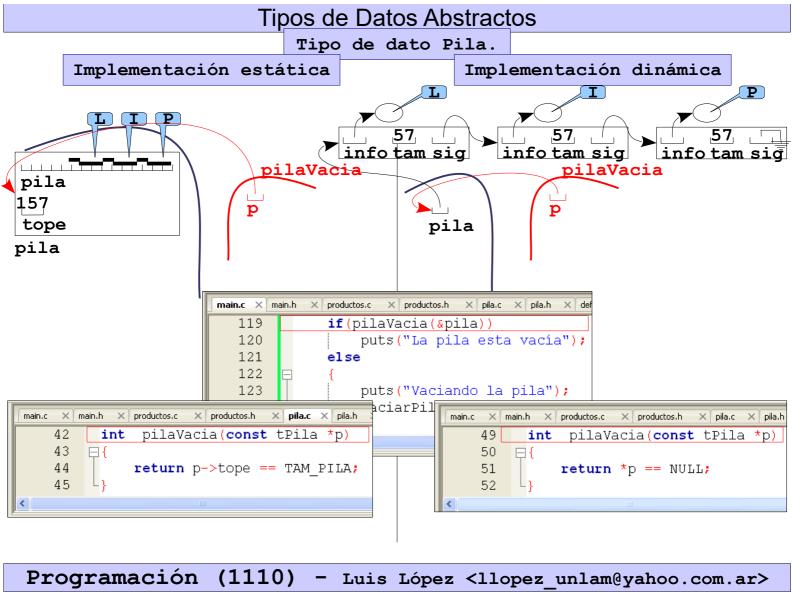


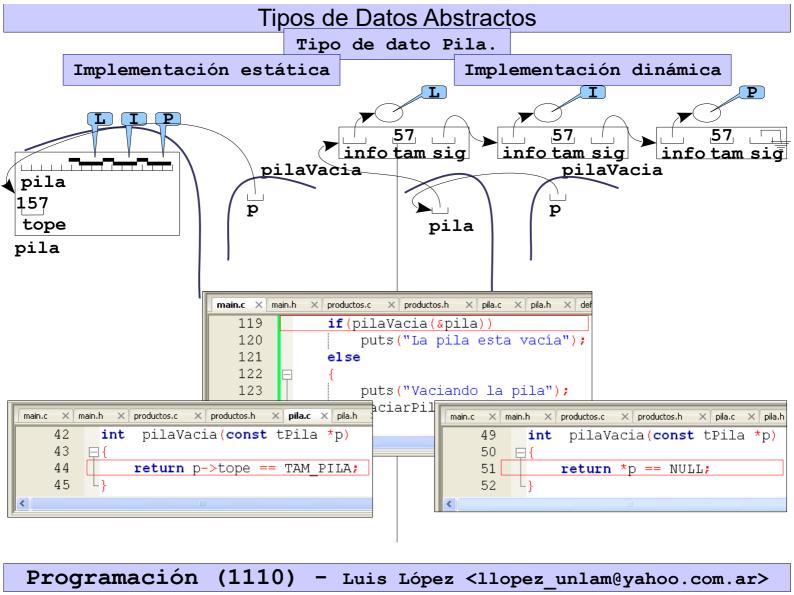


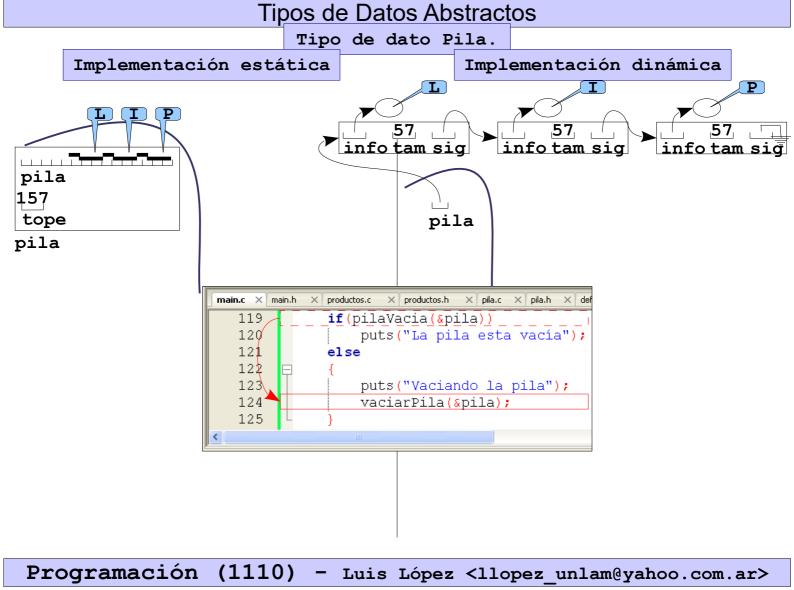
Tipos de Datos Abstractos

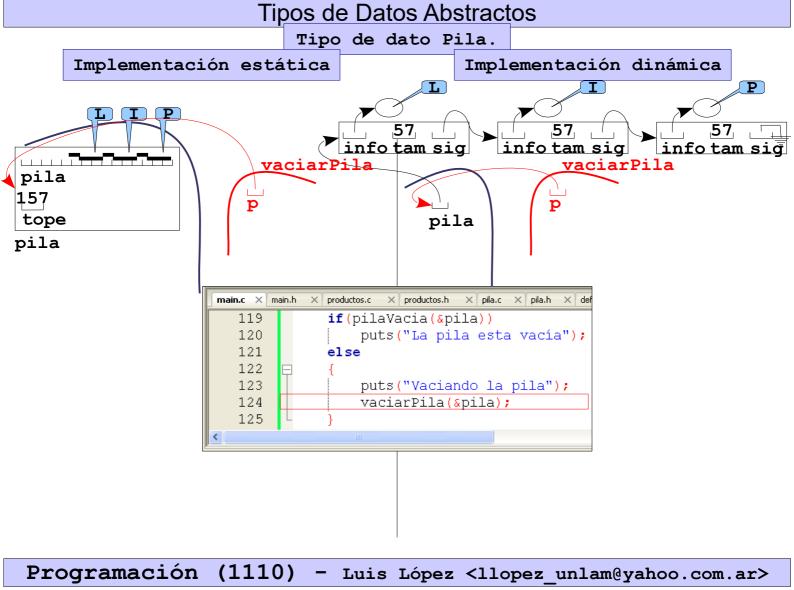


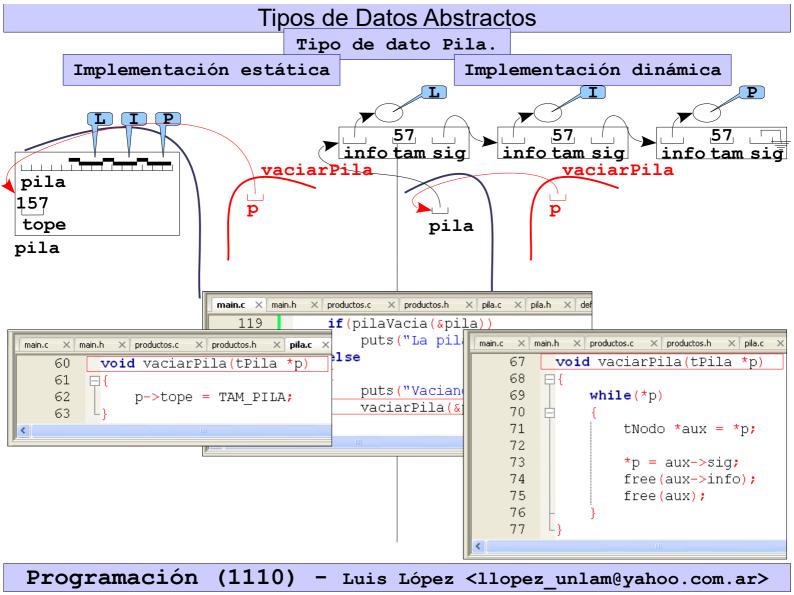
Tipos de Datos Abstractos

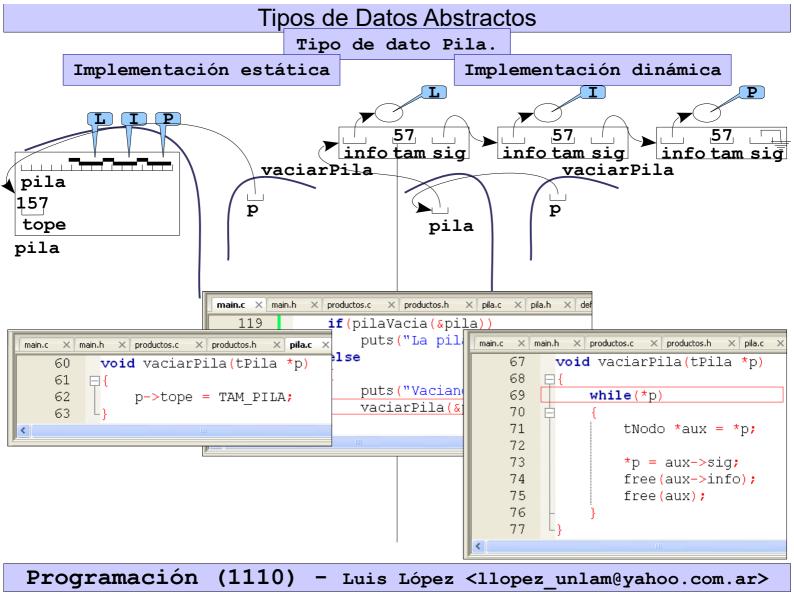


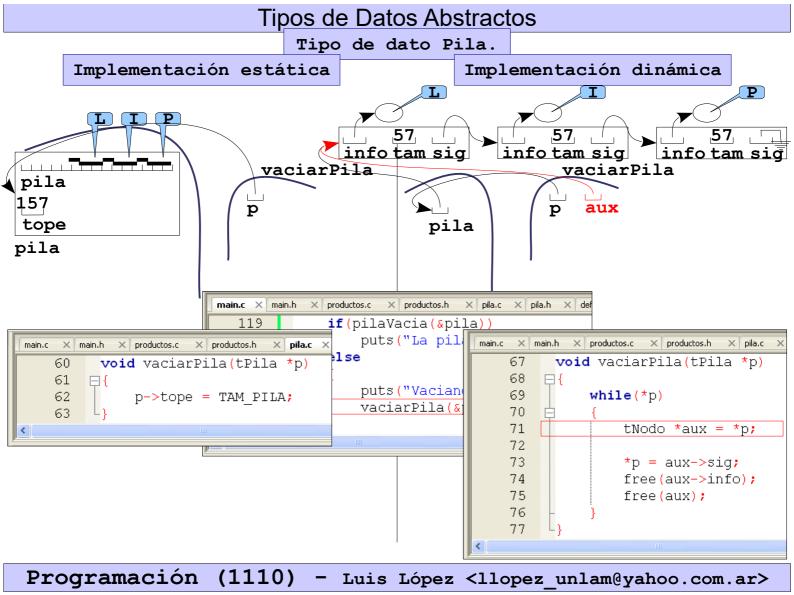


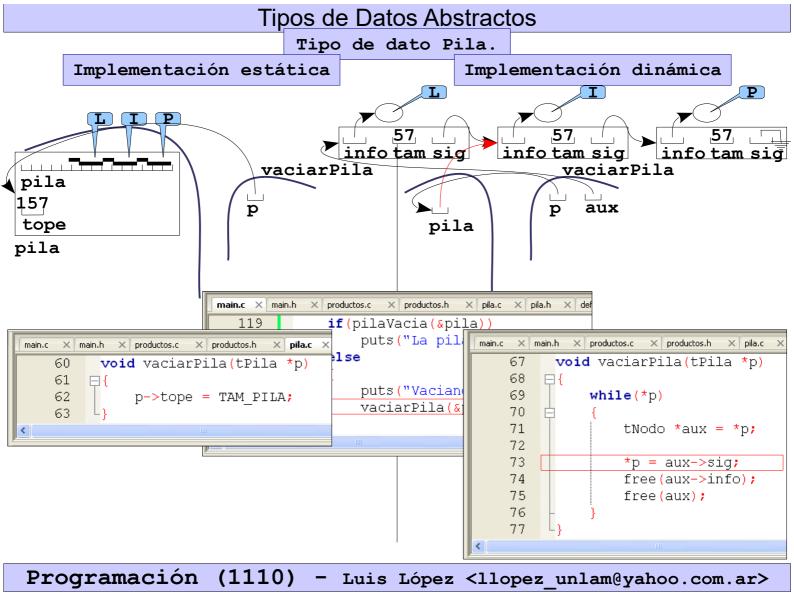


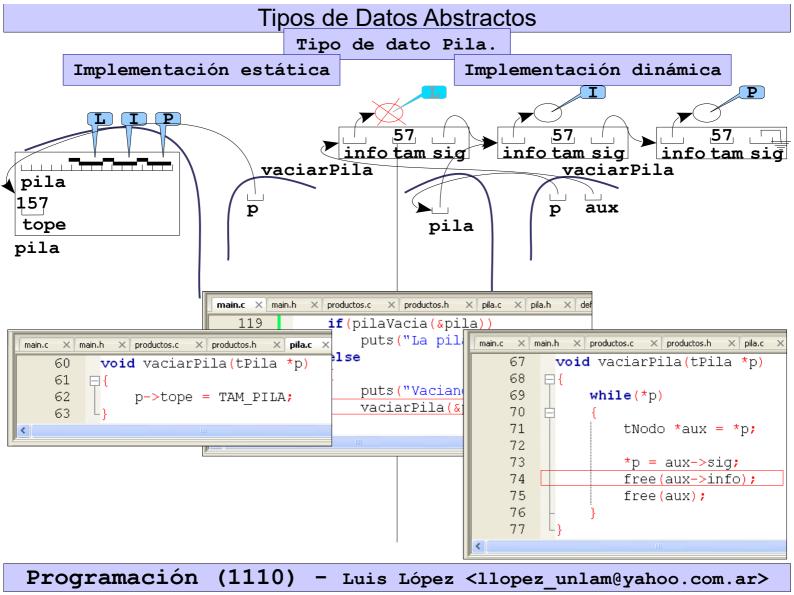


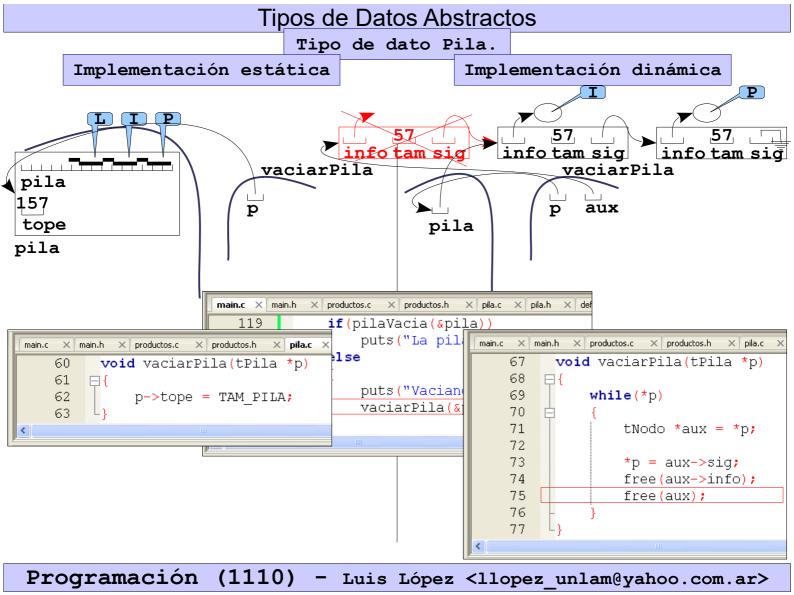


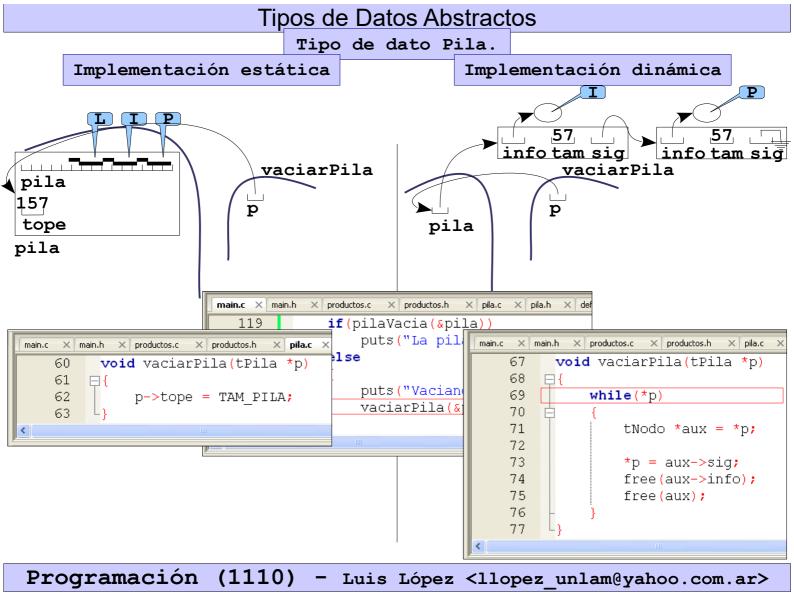


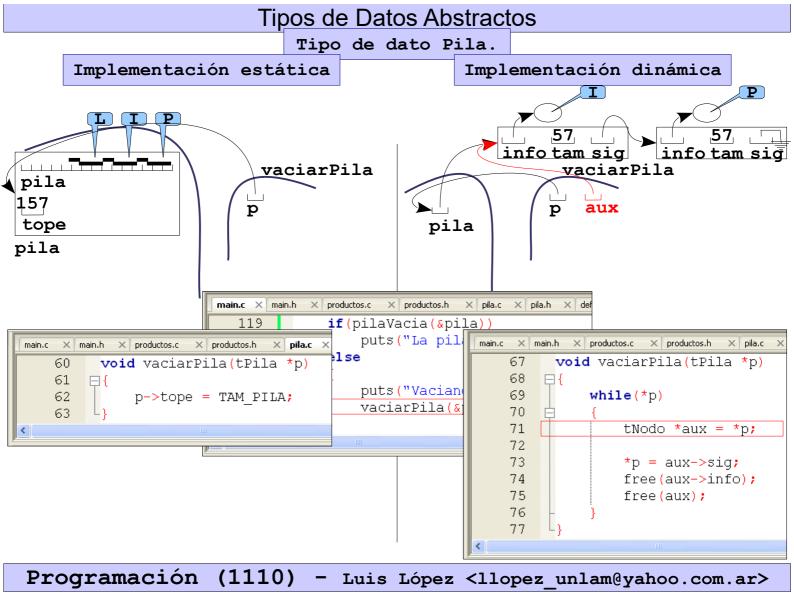


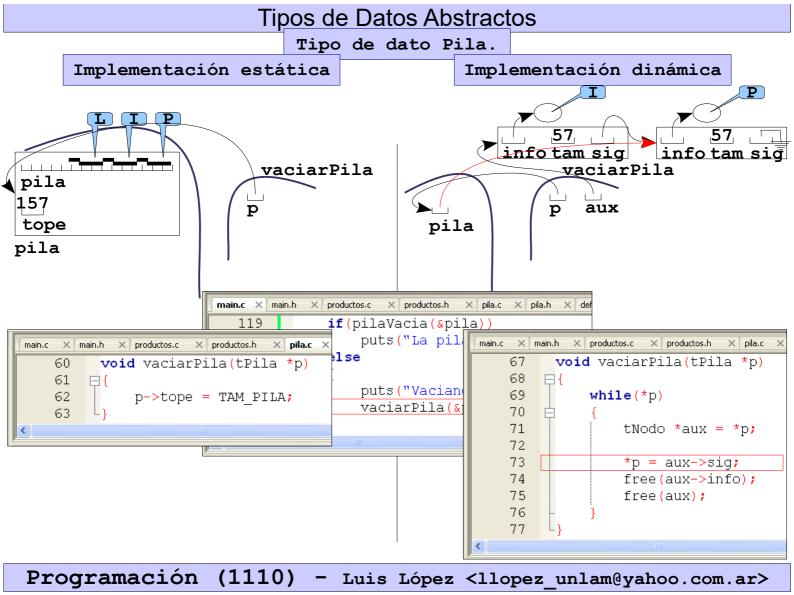


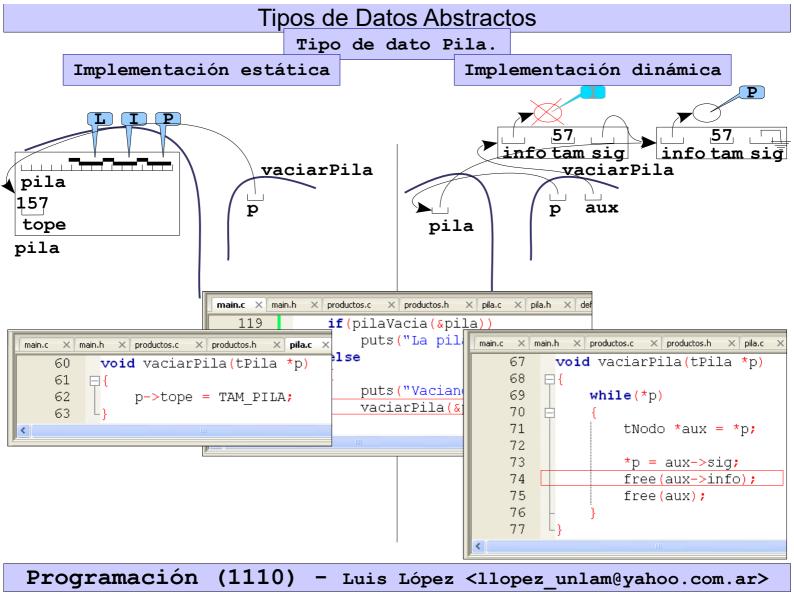


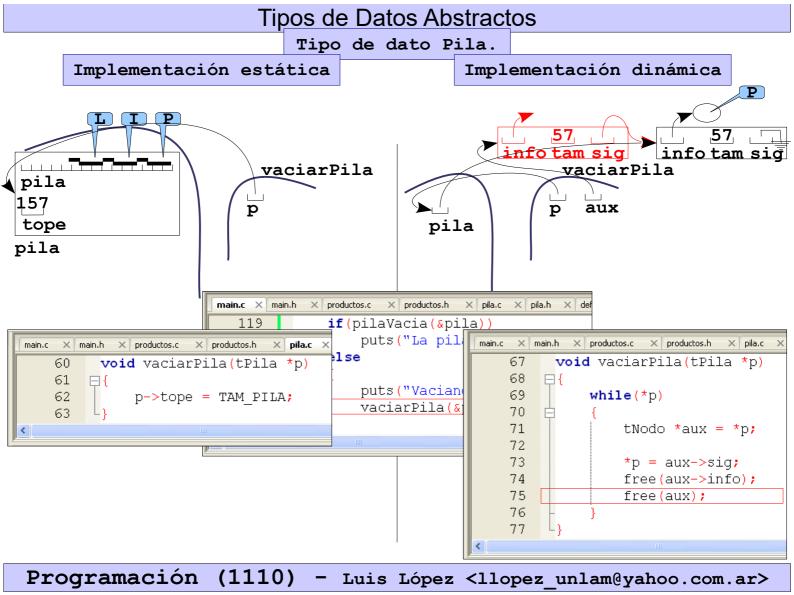


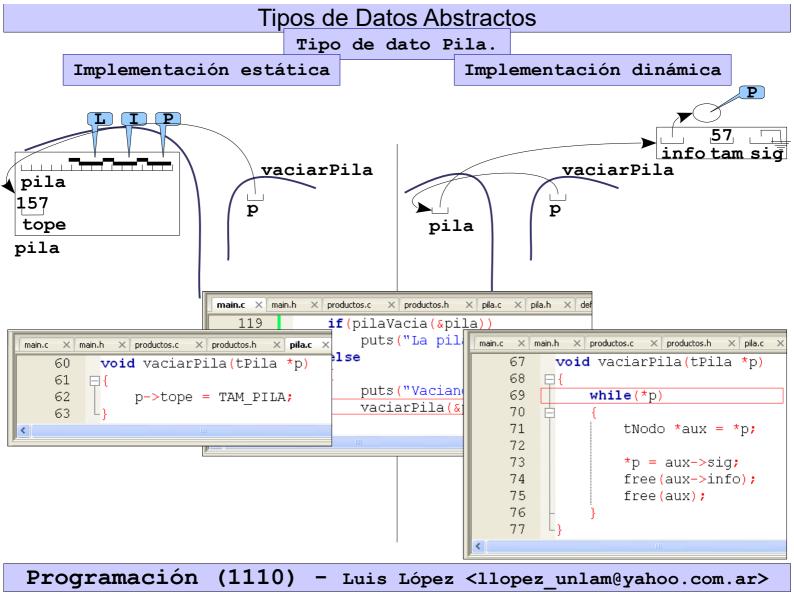


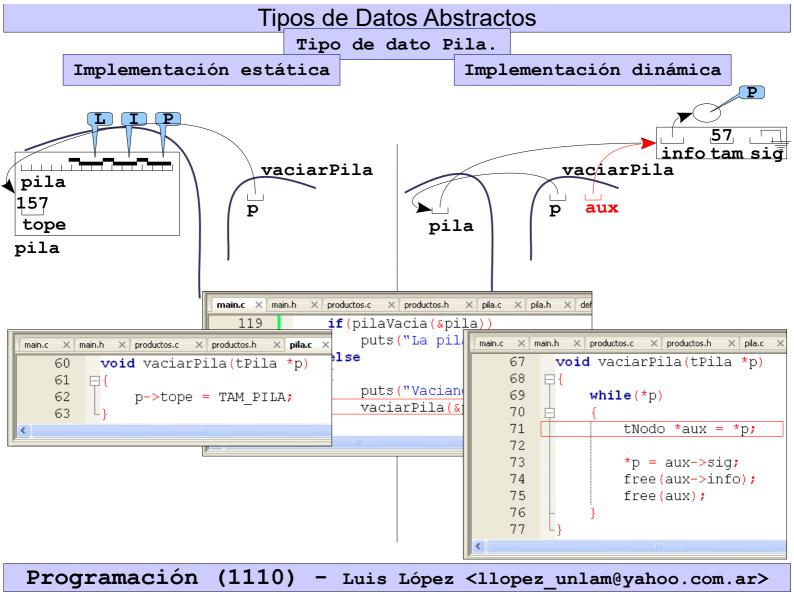


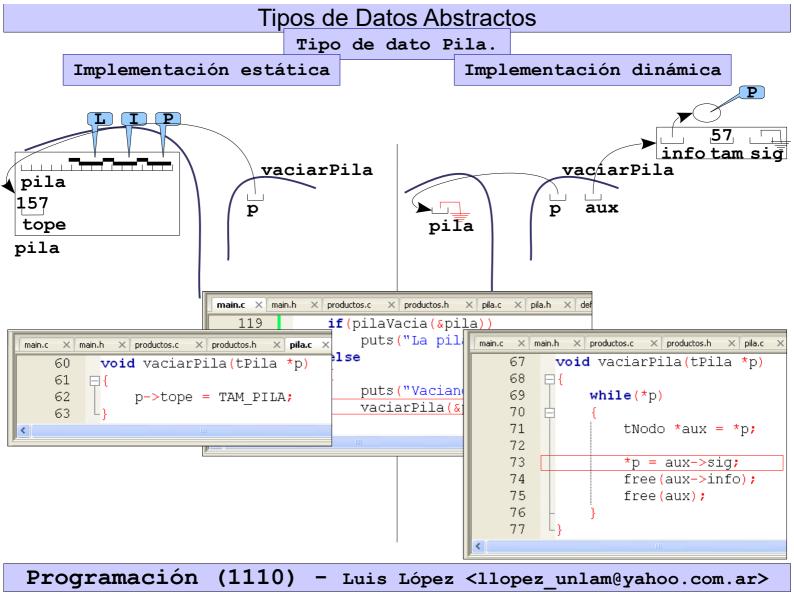


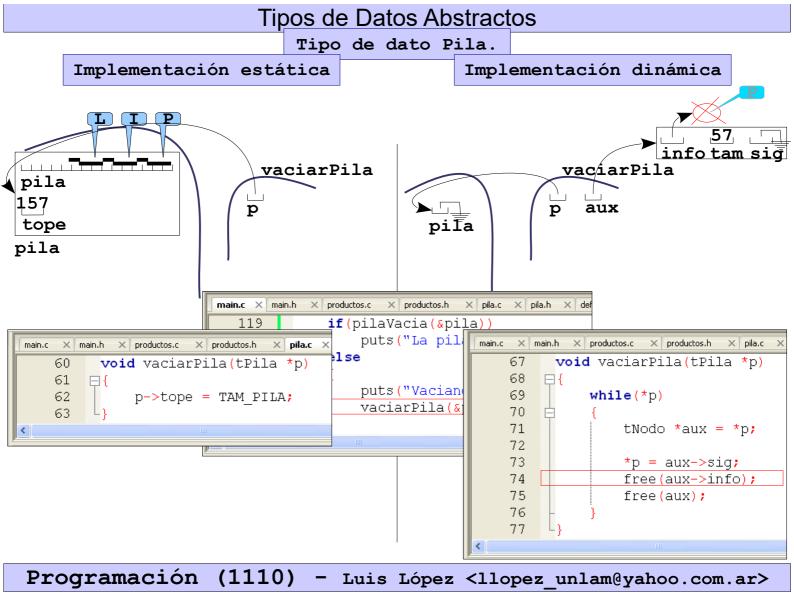


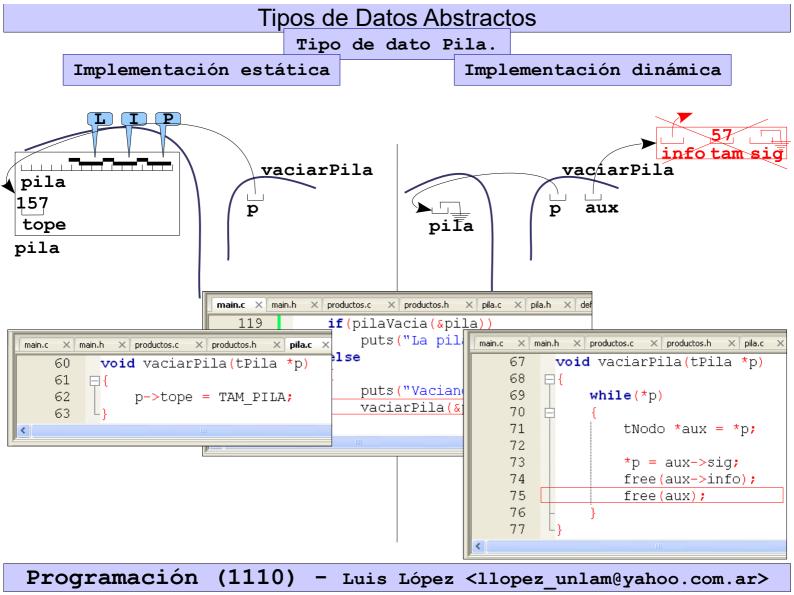


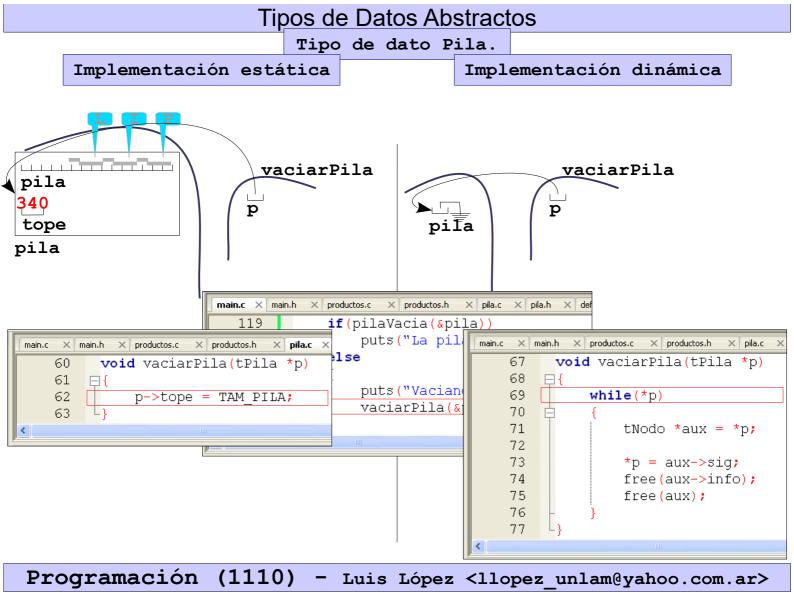


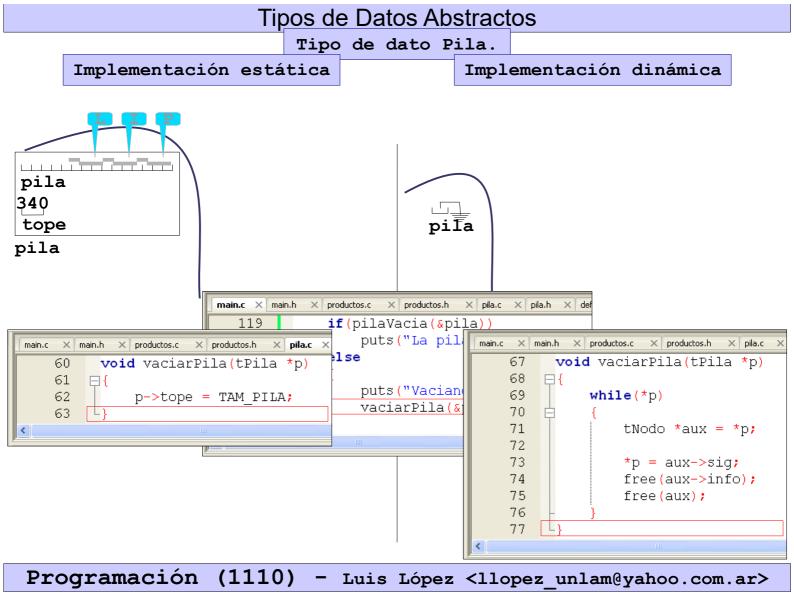


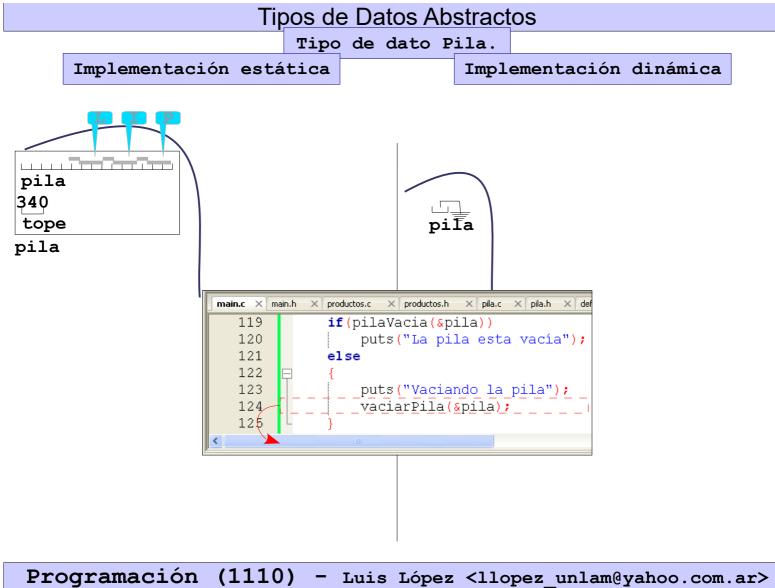




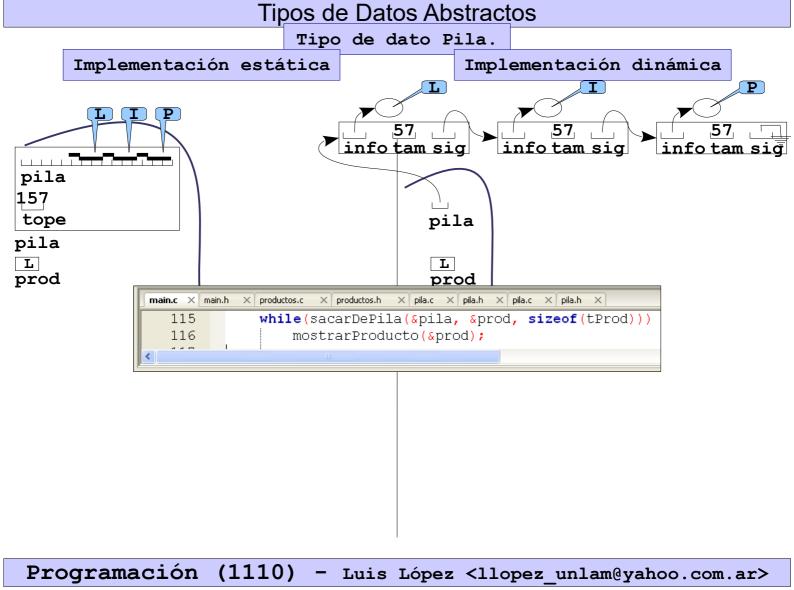












Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica <u>sacarDePila</u> pila 157 tope pila prod cantBytes X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h while(sacarDePila(&pila, &prod, sizeof(tProd))) 115 mostrarProducto(&prod); 116 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 157 tope pila L cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 157 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 53 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) Luis López lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 157 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

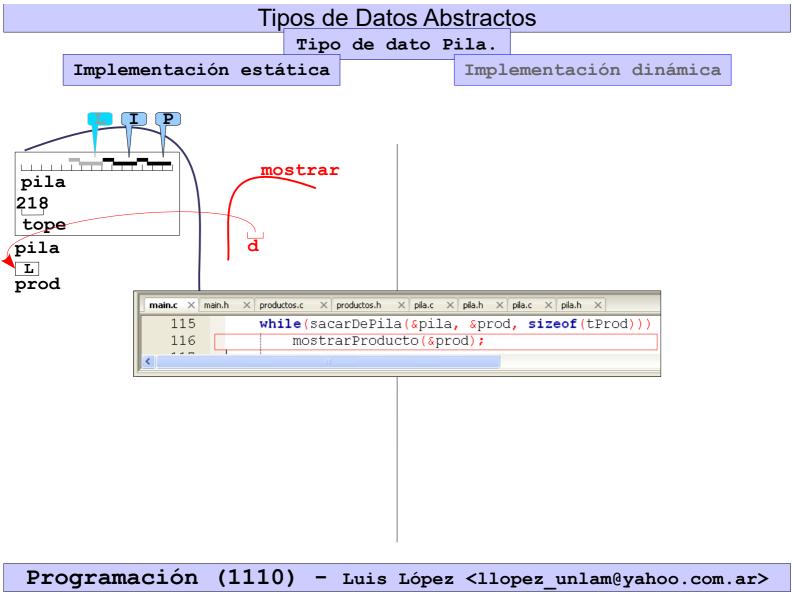
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 161 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 161 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); p->tope += tamInfo; 56 57 return 1; 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 218 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1; 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 218 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1; 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pila 218 tope pila L prod productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h 115 while(sacarDePila(&pila, &prod, sizeof(tProd))) mostrarProducto(&prod); 116 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>



Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pila 218 tope pila L prod X productos.h X pila.c X pila.h X pila.c X pila.h X productos.c 115 while(sacarDePila(&pila, &prod, sizeof(tProd))) mostrarProducto(&prod); Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica <u>sacarDePila</u> pila 218 tope pila cantBytes prod X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h 115 while(sacarDePila(&pila, &prod, sizeof(tProd))) mostrarProducto(&prod); 116 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 218 tope pila L cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 218 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 53 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) Luis López lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 218 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

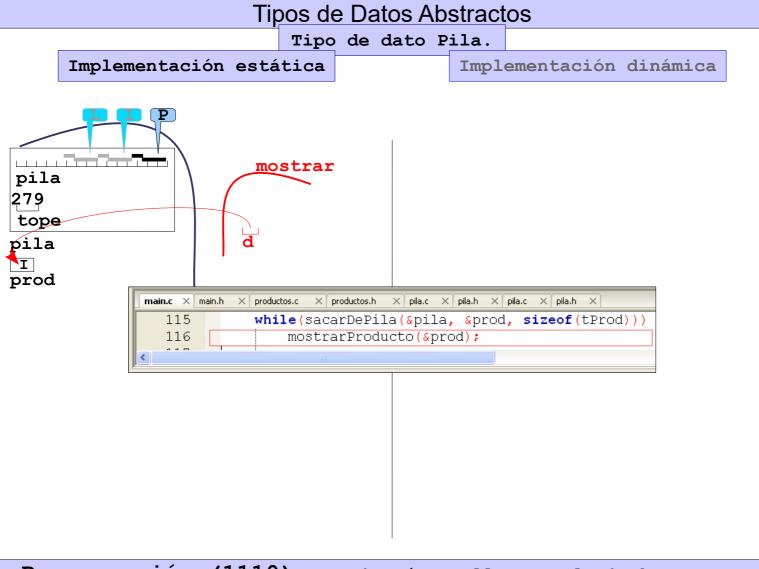
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 222 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); p->tope += tamInfo; 56 57 return 1; 58 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1; 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 279 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1; 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pila 279 tope pila I prod productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h 115 while(sacarDePila(&pila, &prod, sizeof(tProd))) mostrarProducto(&prod); 116 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>



Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica pila 279 tope pila I prod X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X 115 while(sacarDePila(&pila, &prod, sizeof(tProd))) mostrarProducto(&prod); Programación (1110) - Luis López cllopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica <u>sacarDePila</u> pila 279 tope pila prod cantBytes X productos.c X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h 115 while(sacarDePila(&pila, &prod, sizeof(tProd))) mostrarProducto(&prod); 116 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 279 tope pila I cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 279 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 53 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) Luis López lopez unlam@yahoo.com.ar>

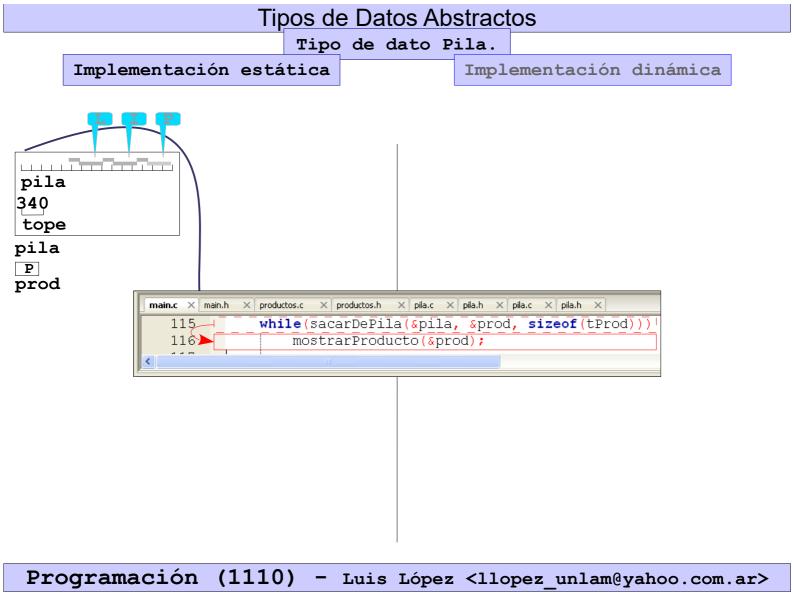
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 279 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

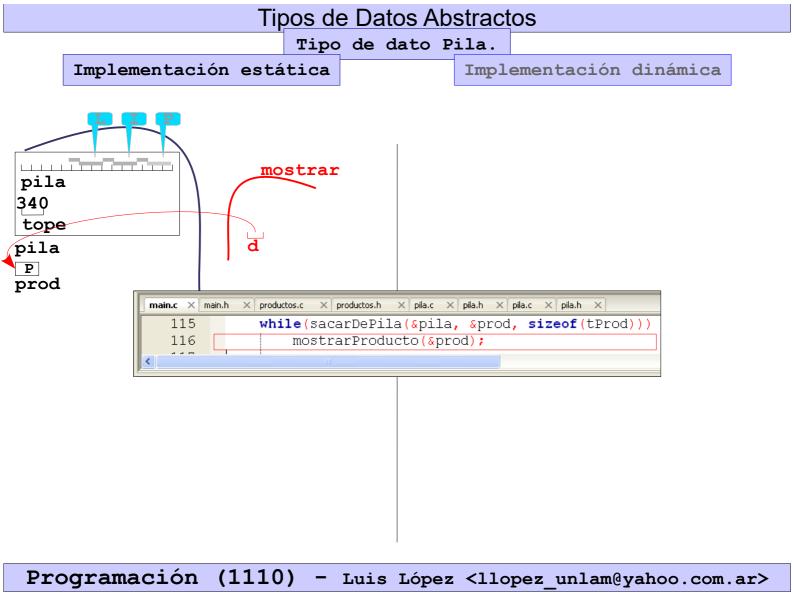
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 283 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

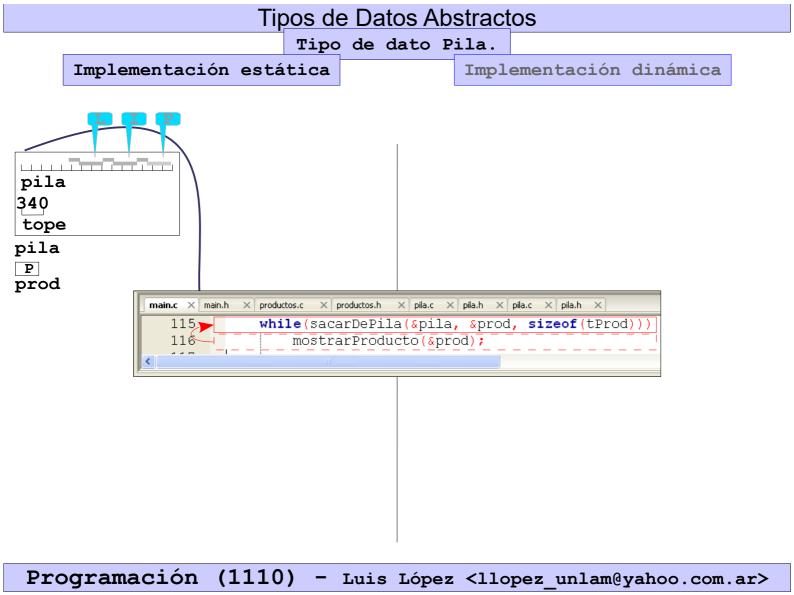
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 283 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); p->tope += tamInfo; 56 57 return 1; 58 Programación (1110) - Luis López Lopez unlam@yahoo.com.ar>

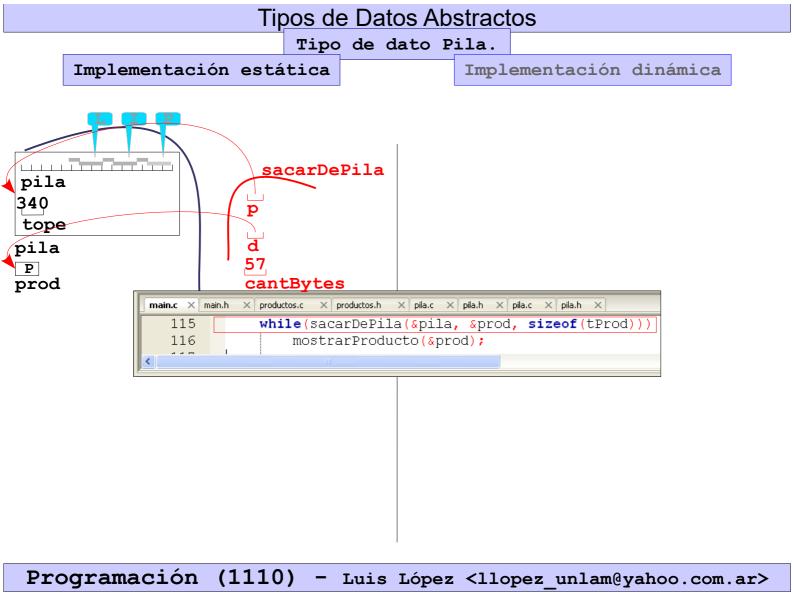
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 340 tamInfo tope pila cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1; 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 340 tamInfo tope pila P cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1; 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>





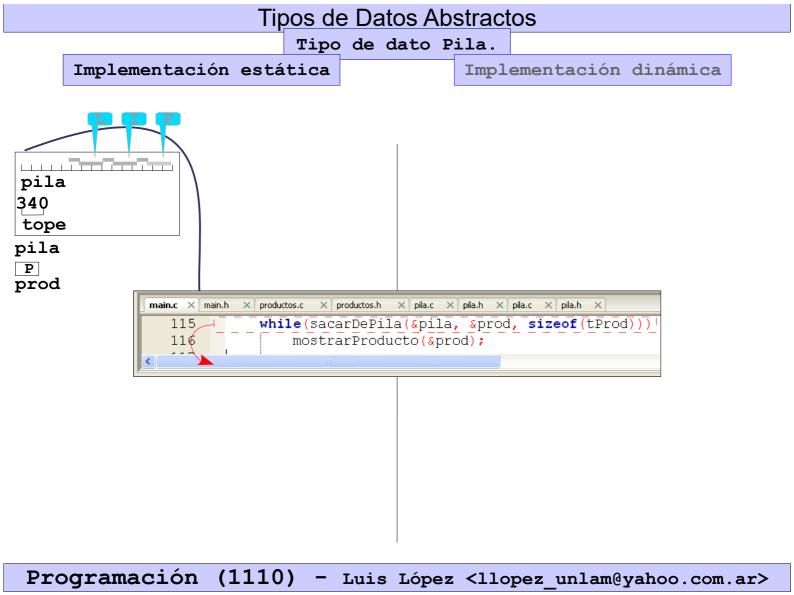




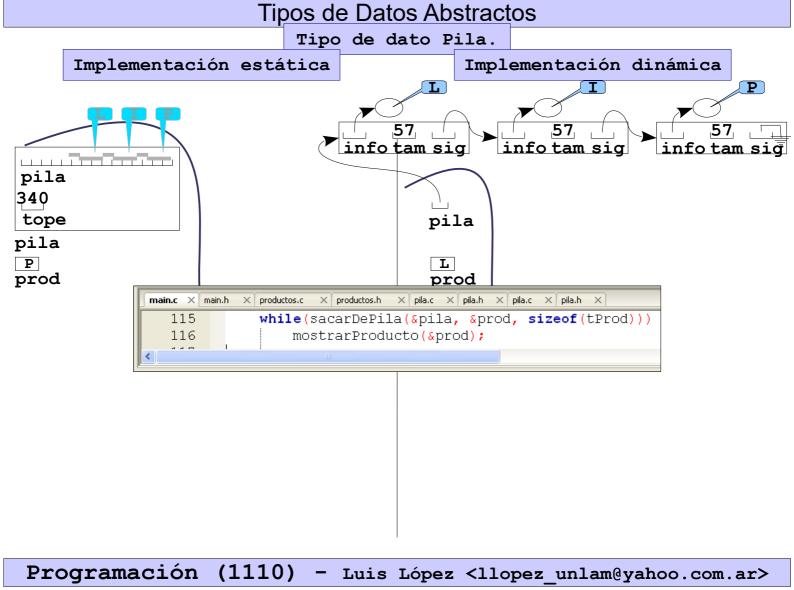
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 340 tope pila P cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

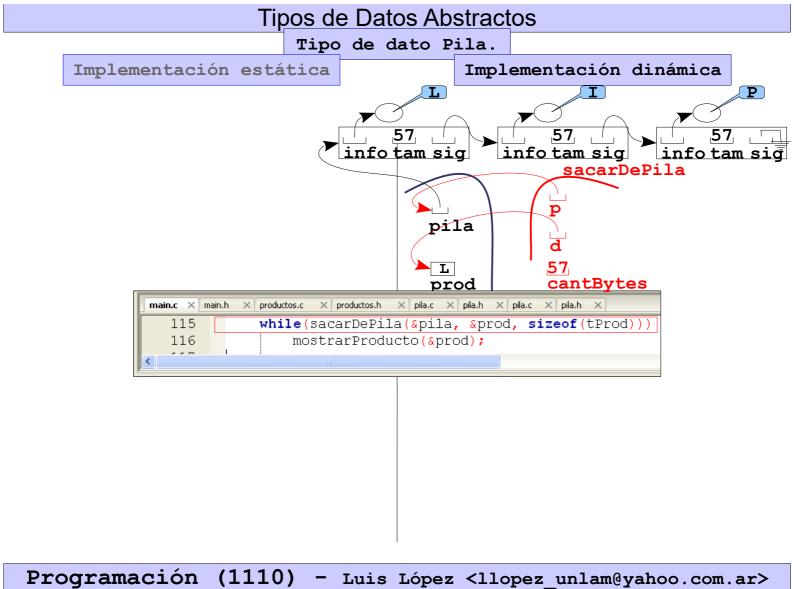
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 340 tamInfo tope pila P cantBytes prod main c X main b X productos c X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 51 if(p->tope == TAM PILA) 52 return 0; memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 53 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) Luis López lopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 340 tamInfo tope pila P cantBytes prod X productos b X pila c X pila b X pila c X pila b × main.h × productos.c × productos.h × pila.c × pila.h × pila.c × pila.h × tProd))) 47 sacarDePila(tPila *p, void *d, unsigned cantBytes) 48 49 unsigned tamInfo; 50 if (p->tope == TAM PILA) 51 52 return 0; 53 memcpy(&tamInfo, p->pila + p->tope, sizeof(unsigned)); 54 p->tope += sizeof(unsigned); 55 memcpy(d, p->pila + p->tope, minimo(cantBytes, tamInfo)); 56 p->tope += tamInfo; 57 return 1: 58 Programación (1110) Luis López com.ar>



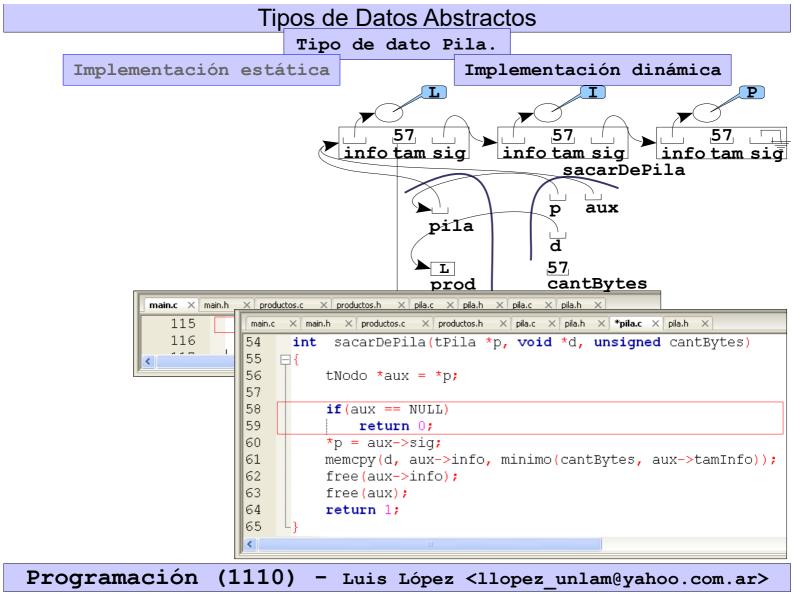


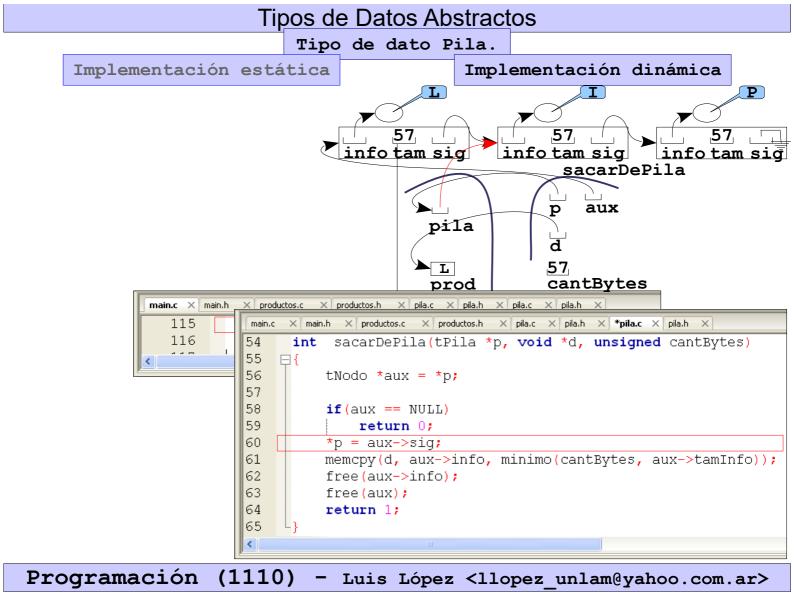


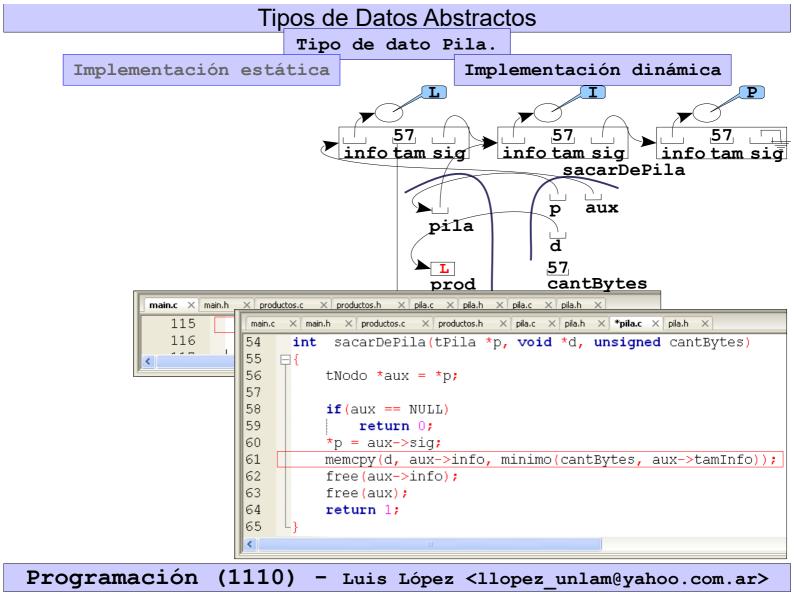


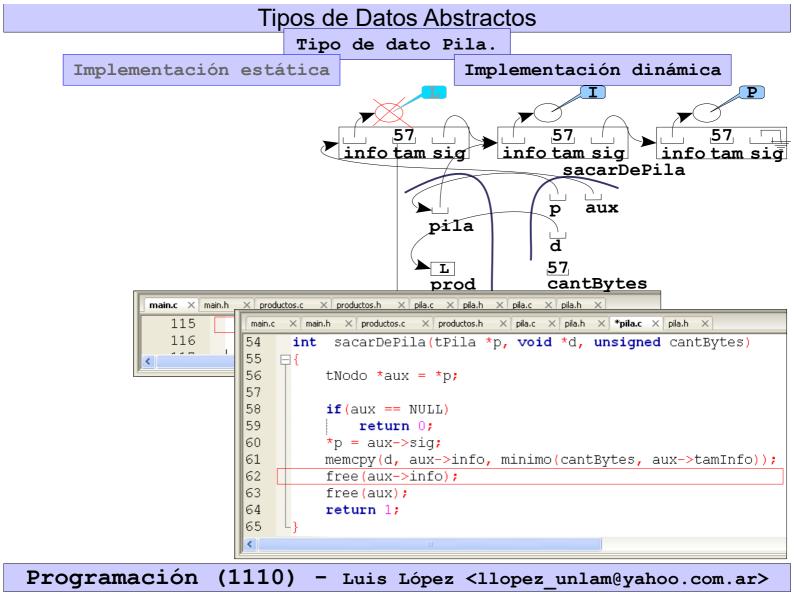
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam sig info tam sia sacarDePila pila cantBytes prod X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h × productos.c 115 × main.h × productos.c × productos.h × pila.c × pila.h × *pila.c × pila.h × 116 54 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 55 56 tNodo *aux = *p; 57 58 if (aux == NULL) 59 return 0: 60 *p = aux->siq;61 memcpy(d, aux->info, minimo(cantBytes, aux->tamInfo)); 62 free (aux->info); 63 free (aux); 64 return 1; 65 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam sig info tam sig info tam sia sacarDePila aux pila 57 cantBytes prod X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h × productos.c 115 × main.h × productos.c × productos.h × pila.c × pila.h × *pila.c × pila.h × 116 54 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 55 56 tNodo *aux = *p; 57 58 if (aux == NULL) 59 return 0: 60 *p = aux->siq;61 memcpy(d, aux->info, minimo(cantBytes, aux->tamInfo)); 62 free (aux->info); 63 free (aux); 64 return 1; 65 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

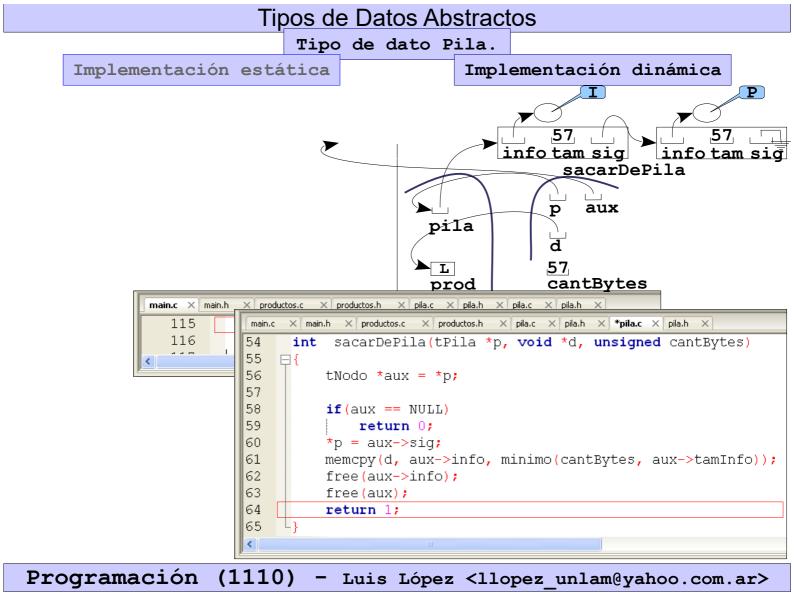


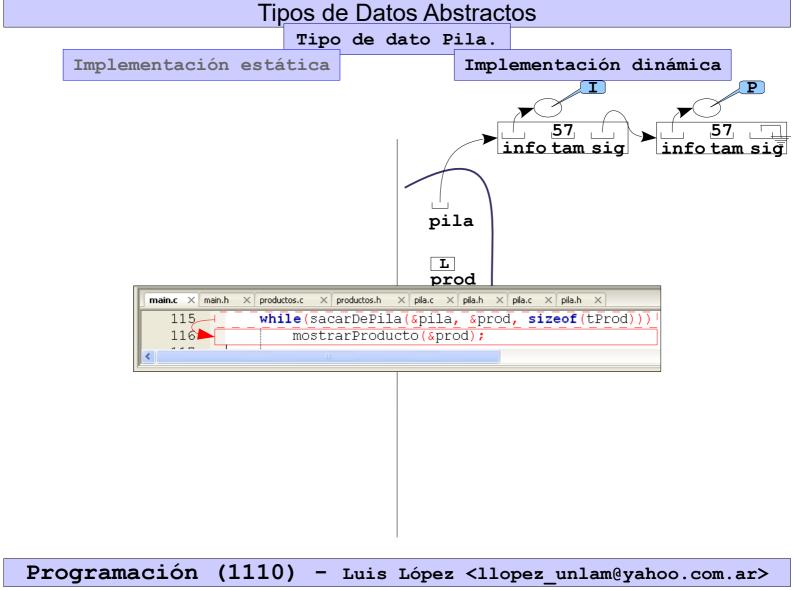


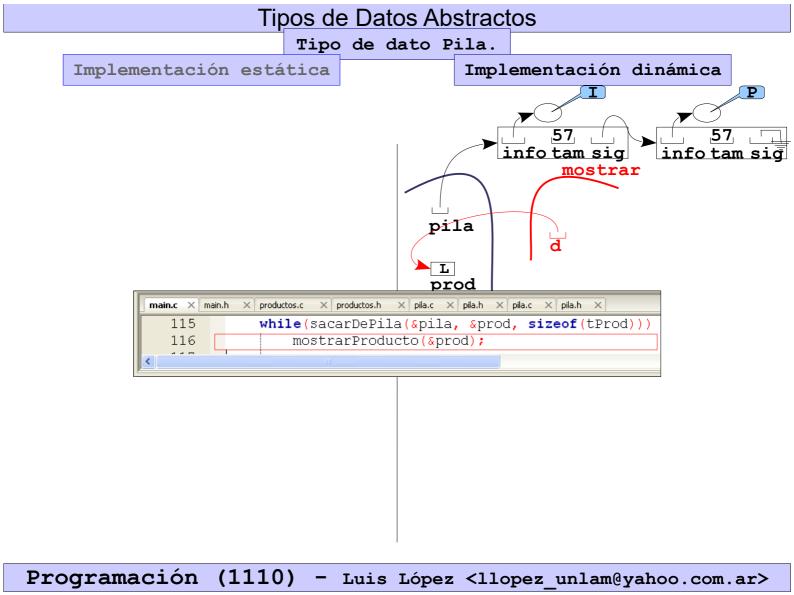


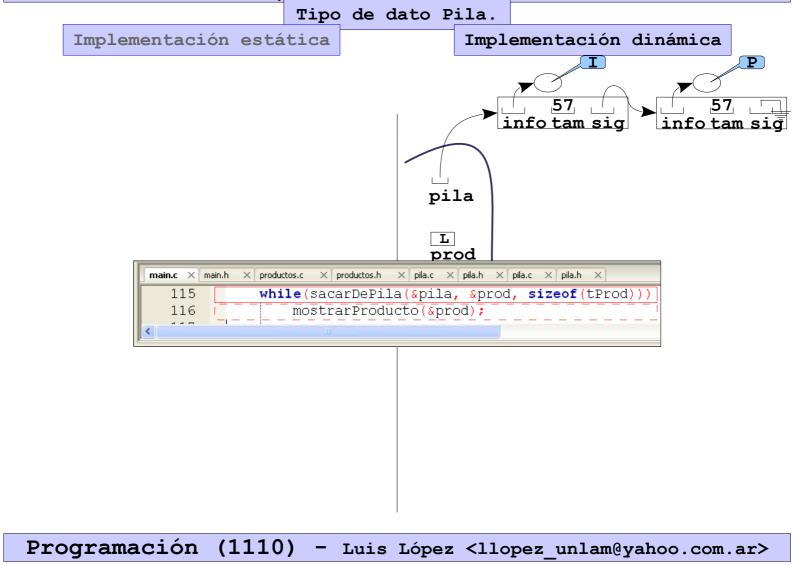


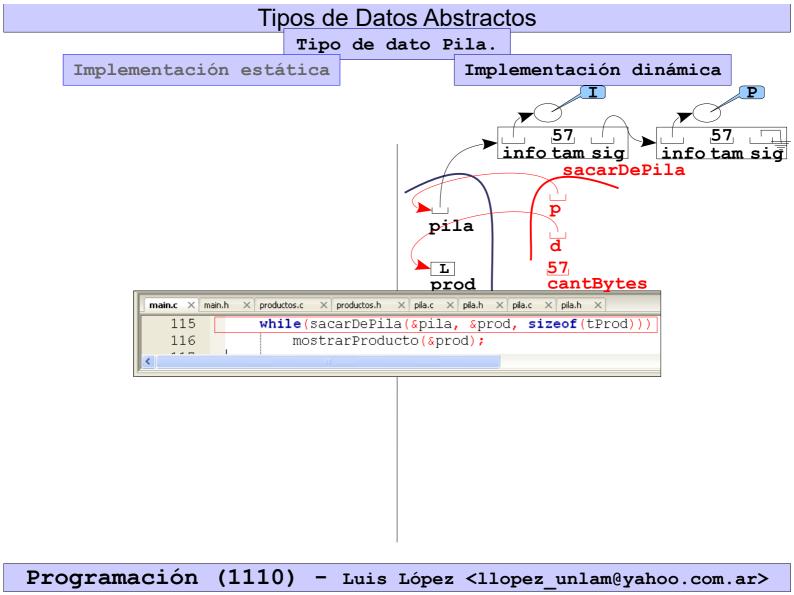
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 57 info tam sig info tam sig info tam sia sacarDePila aux pila 57 cantBytes prod X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h × productos.c 115 × main.h × productos.c × productos.h × pila.c × pila.h × *pila.c × pila.h × 116 54 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 55 56 tNodo *aux = *p; 57 58 if (aux == NULL) 59 return 0: 60 *p = aux->siq;61 memcpy(d, aux->info, minimo(cantBytes, aux->tamInfo)); 62 free(aux->info); 63 free (aux); 64 return 1; 65 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

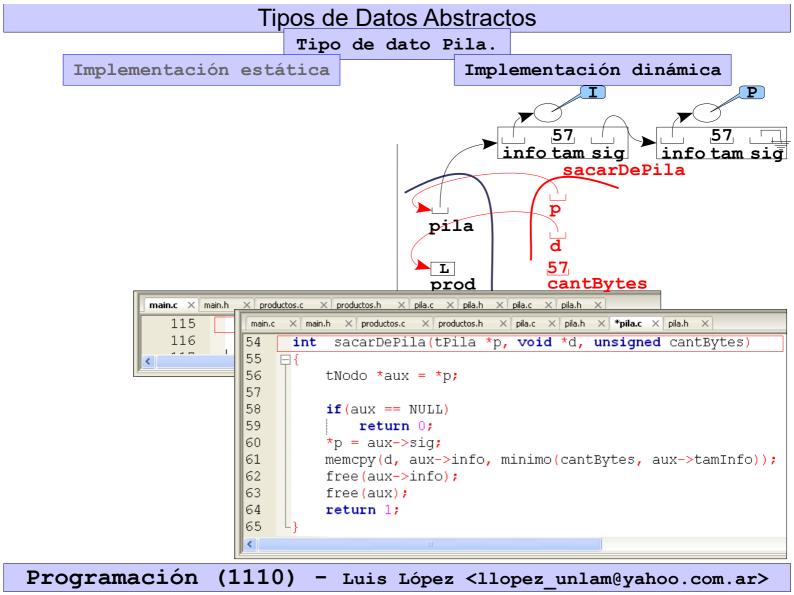


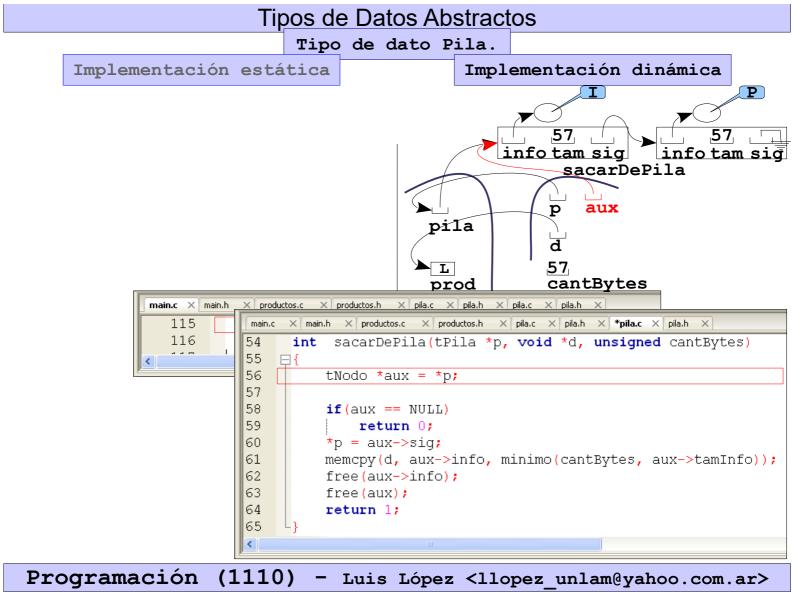


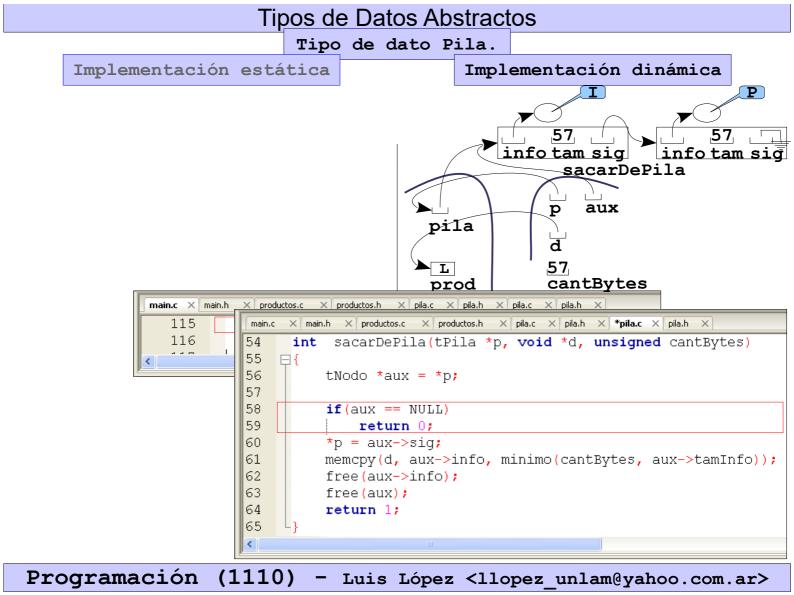


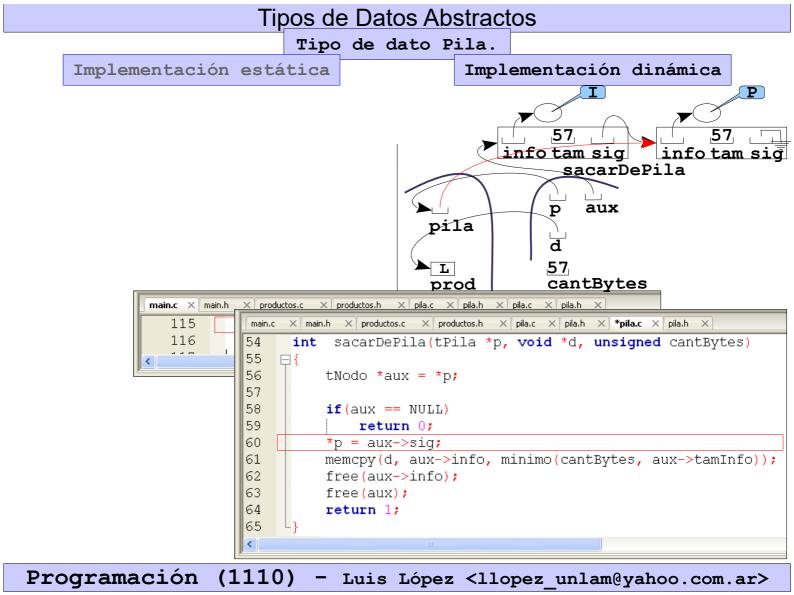


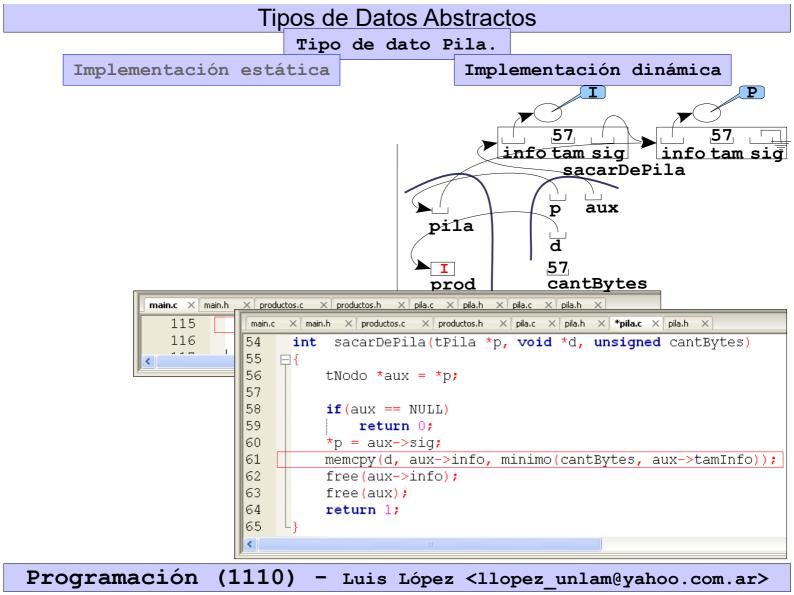


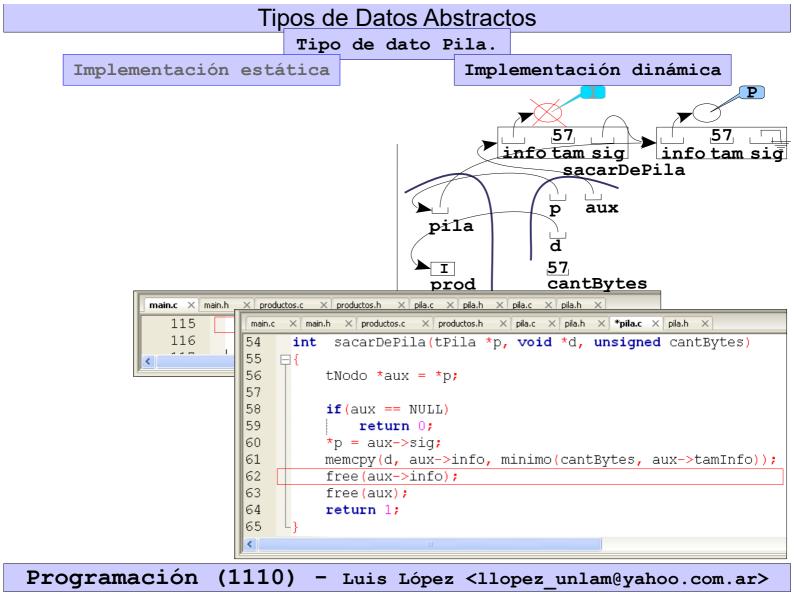


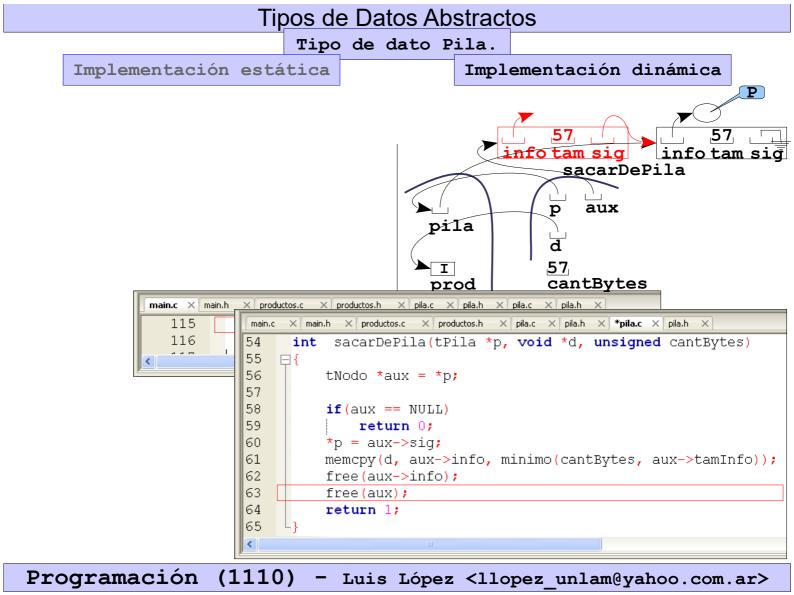


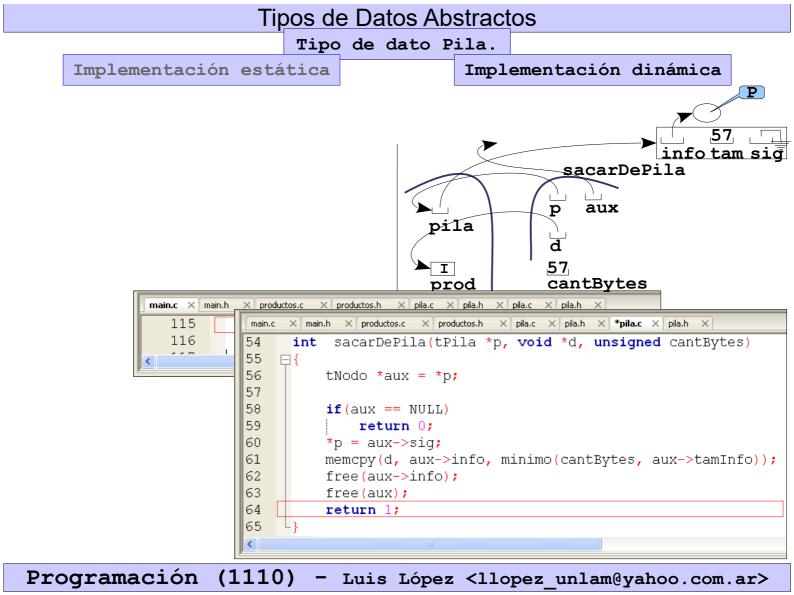


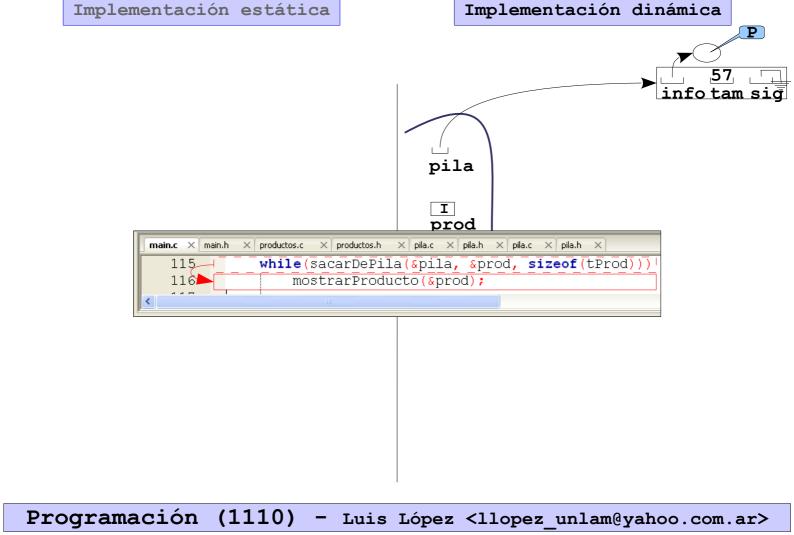




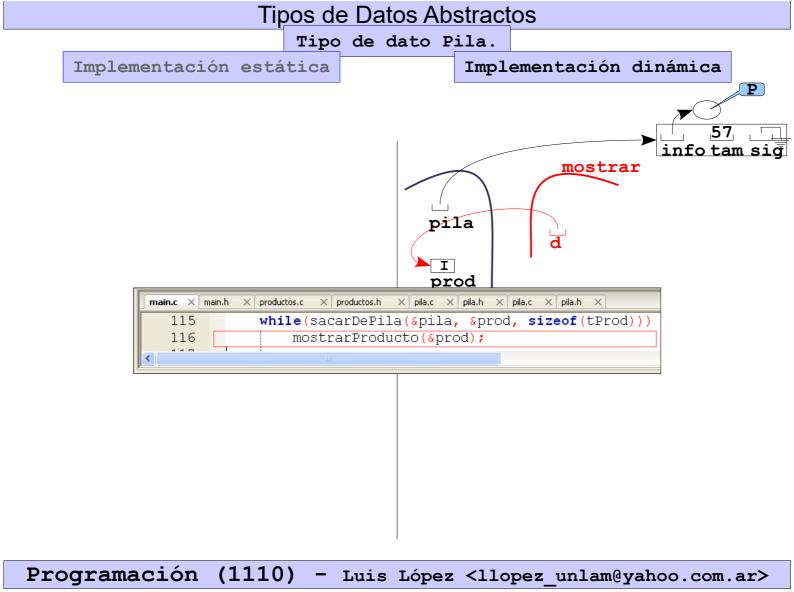


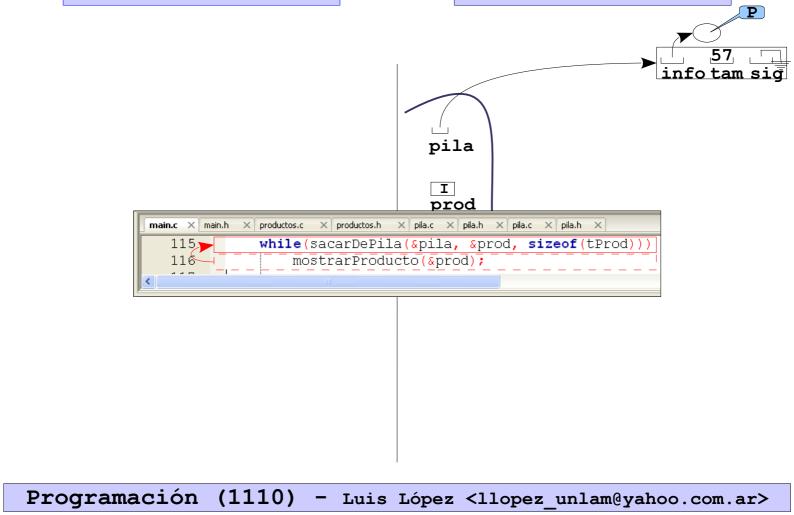






Tipos de Datos Abstractos
Tipo de dato Pila.

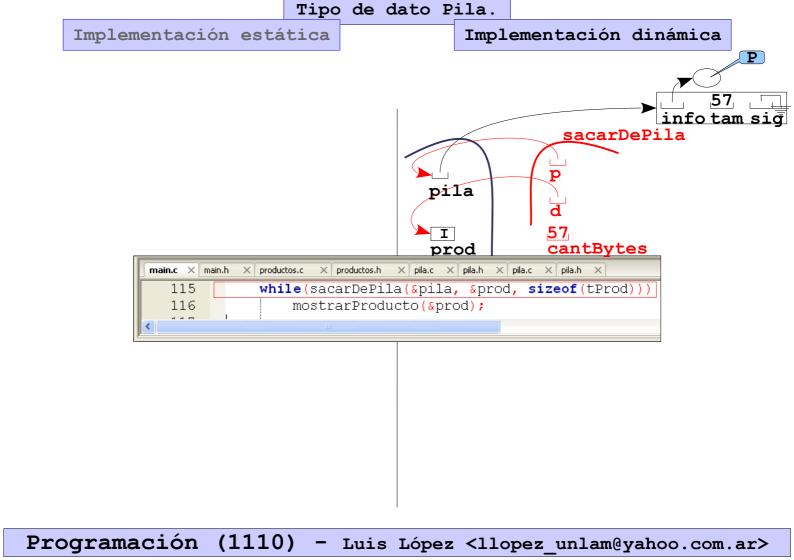




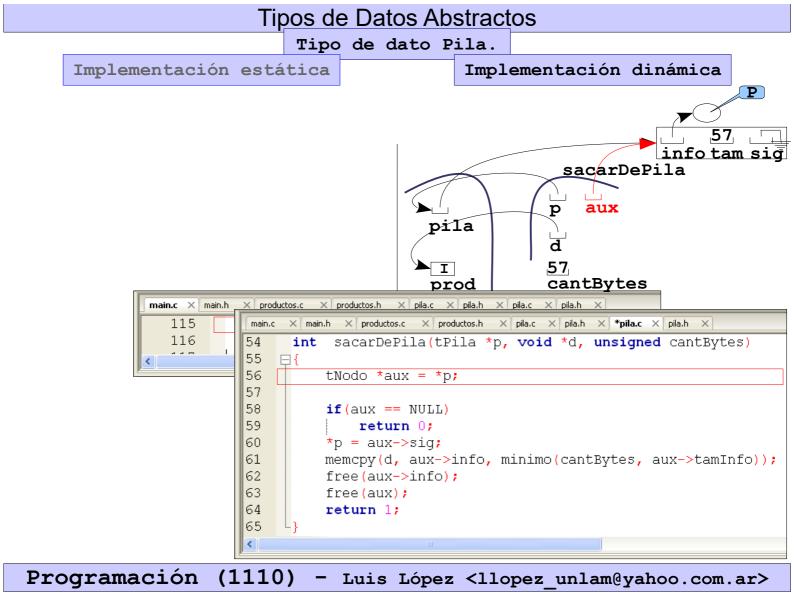
Tipos de Datos Abstractos
Tipo de dato Pila.

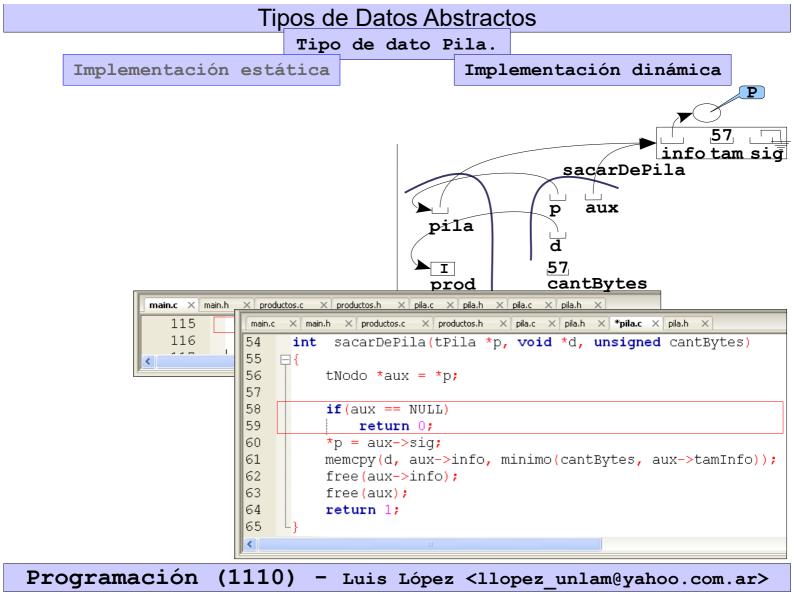
Implementación dinámica

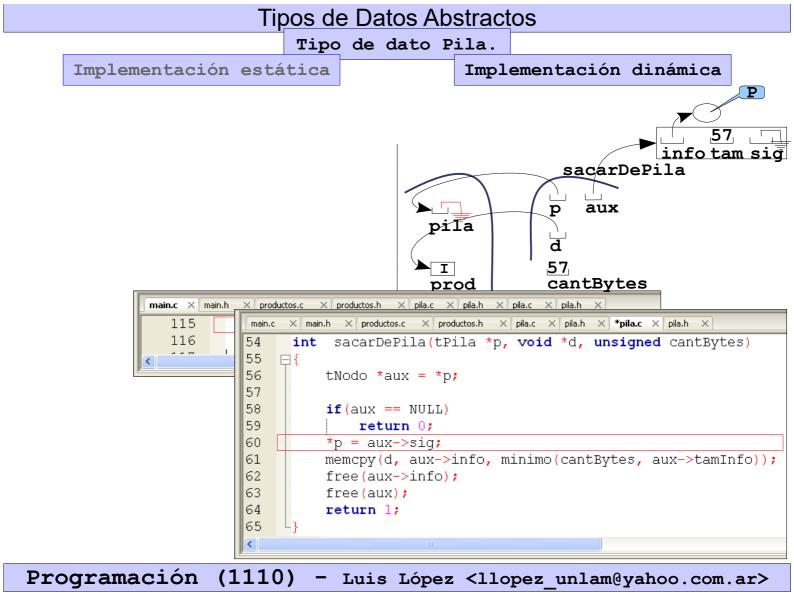
Implementación estática

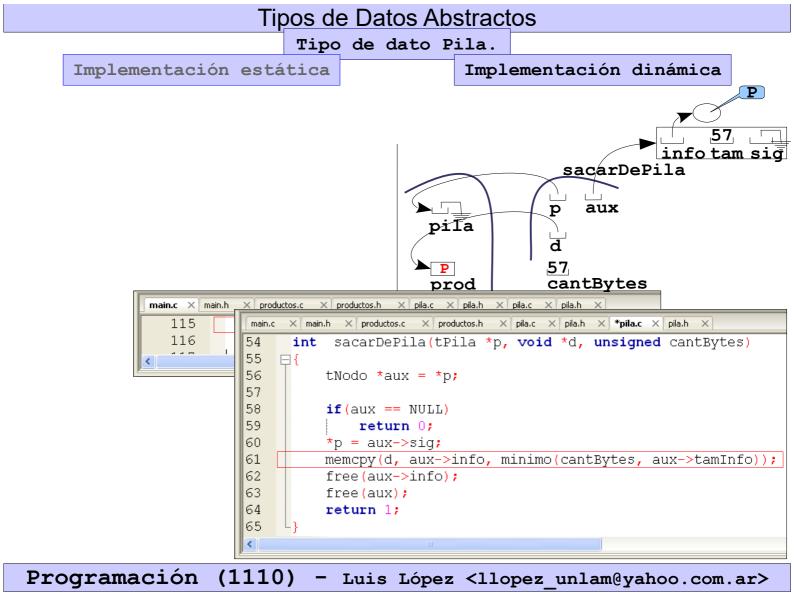


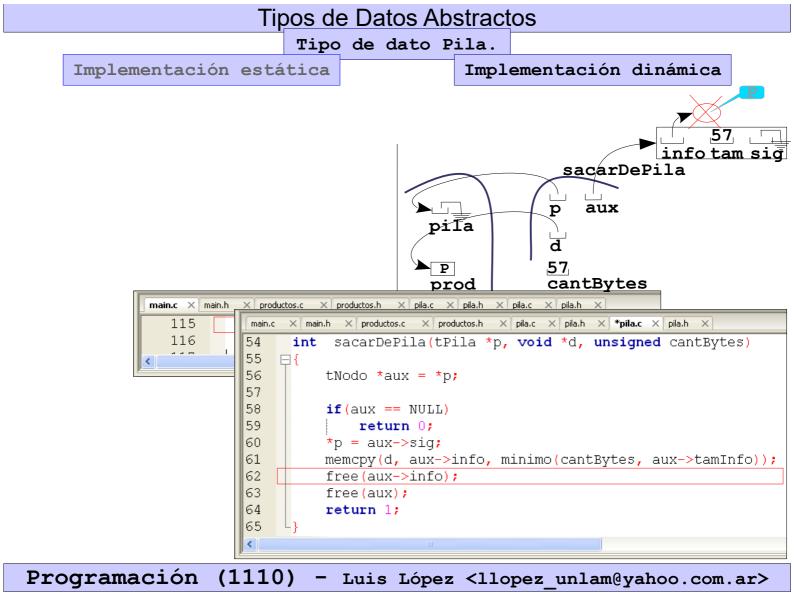
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica 57 info tam siq <u>sacarDePila</u> pila cantBytes prod X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h × productos.c 115 X main.h X productos.c X productos.h X pila.c X pila.h X *pila.c X 116 54 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 55 56 tNodo *aux = *p; 57 58 if (aux == NULL) 59 return 0: 60 *p = aux->siq;61 memcpy(d, aux->info, minimo(cantBytes, aux->tamInfo)); 62 free (aux->info); 63 free (aux); 64 return 1; 65 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>









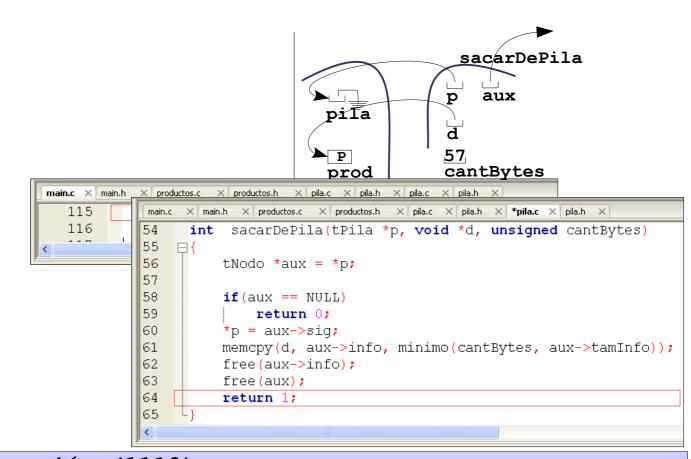


Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica info tam sig sacarDePila aux pila 57 cantBytes prod X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h × productos.c 115 × main.h × productos.c × productos.h × pila.c × pila.h × *pila.c × pila.h × 116 54 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 55 56 tNodo *aux = *p; 57 58 if (aux == NULL) 59 return 0: 60 *p = aux->siq;61 memcpy(d, aux->info, minimo(cantBytes, aux->tamInfo)); 62 free(aux->info); 63 free (aux); 64 return 1; 65 Programación (1110) - Luis López <llopez unlam@yahoo.com.ar>

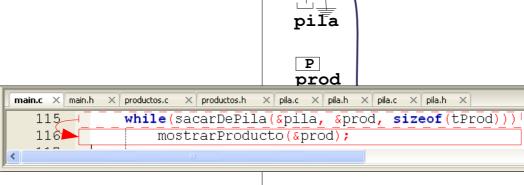
Tipos de Datos Abstractos Tipo de dato Pila.

Tipo de dato Fila.

Implementación estática

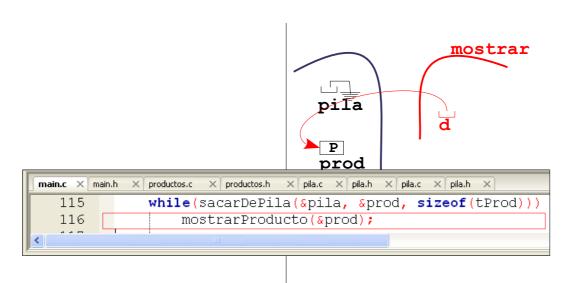


Tipo de dato Pila. Implementación estática Implementación dinámica

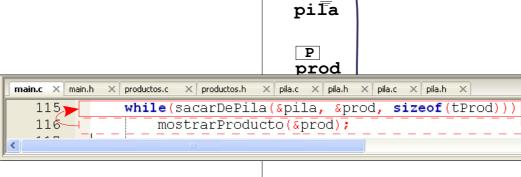


Tipo de dato Pila.

Implementación estática

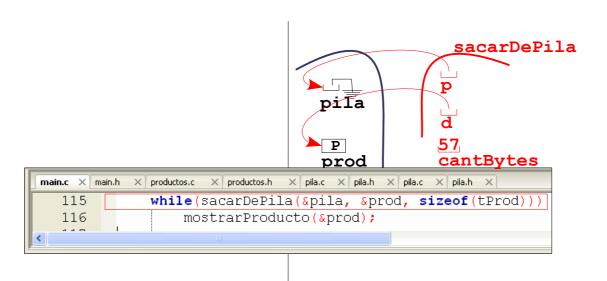


Tipo de dato Pila. Implementación estática Implementación dinámica

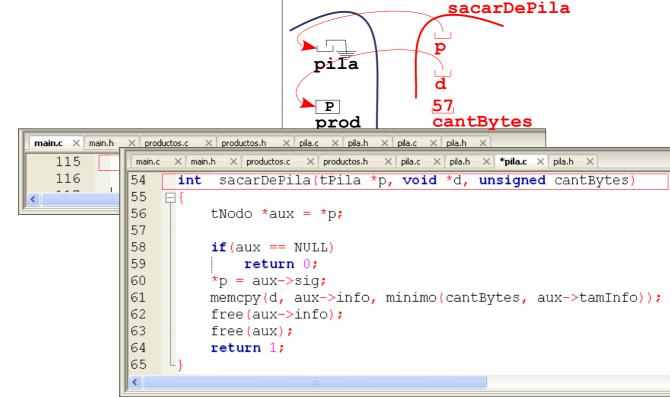


Tipo de dato Pila.

Implementación estática



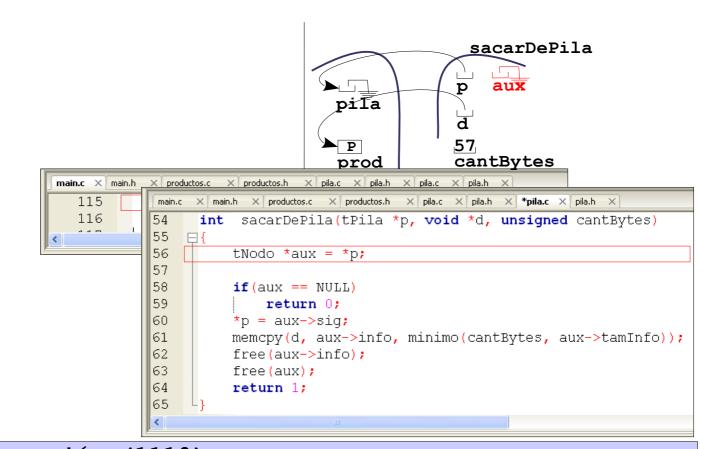
Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila



Tipo de dato Pila.

Implementación estática

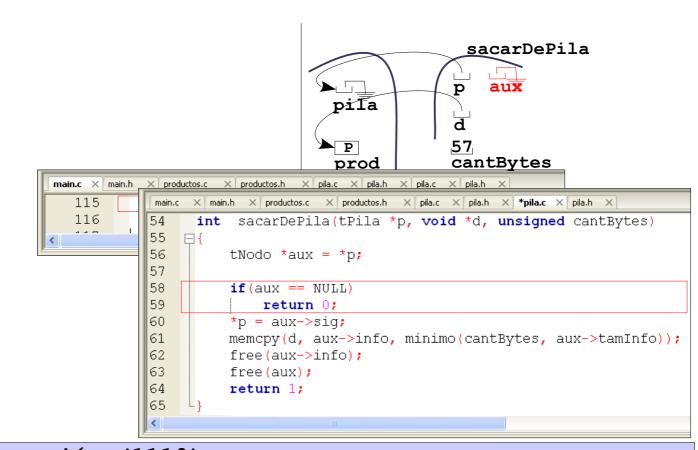
Implementación dinámica



Programación (1110) - Luis López < llopez _unlam@yahoo.com.ar>

Tipo de dato Pila.

Implementación estática



Tipos de Datos Abstractos Tipo de dato Pila. Implementación estática Implementación dinámica sacarDePila pila 57 Р cantBytes prod X productos.h X pila.c X pila.h X pila.c X pila.h X main.c × main.h × productos.c 115 × main.h × productos.c × productos.h × pila.c × pila.h × *pila.c × pila.h × 116 54 int sacarDePila(tPila *p, void *d, unsigned cantBytes) 55 56 tNodo *aux = *p; 57 if (aux == NULL) 58 59 return 0; 60 *p = aux->siq;

free (aux->info);

free (aux);

return 1;

memcpy(d, aux->info, minimo(cantBytes, aux->tamInfo));

61

62

63

64

65

Tipo de dato Pila.

Implementación estática

