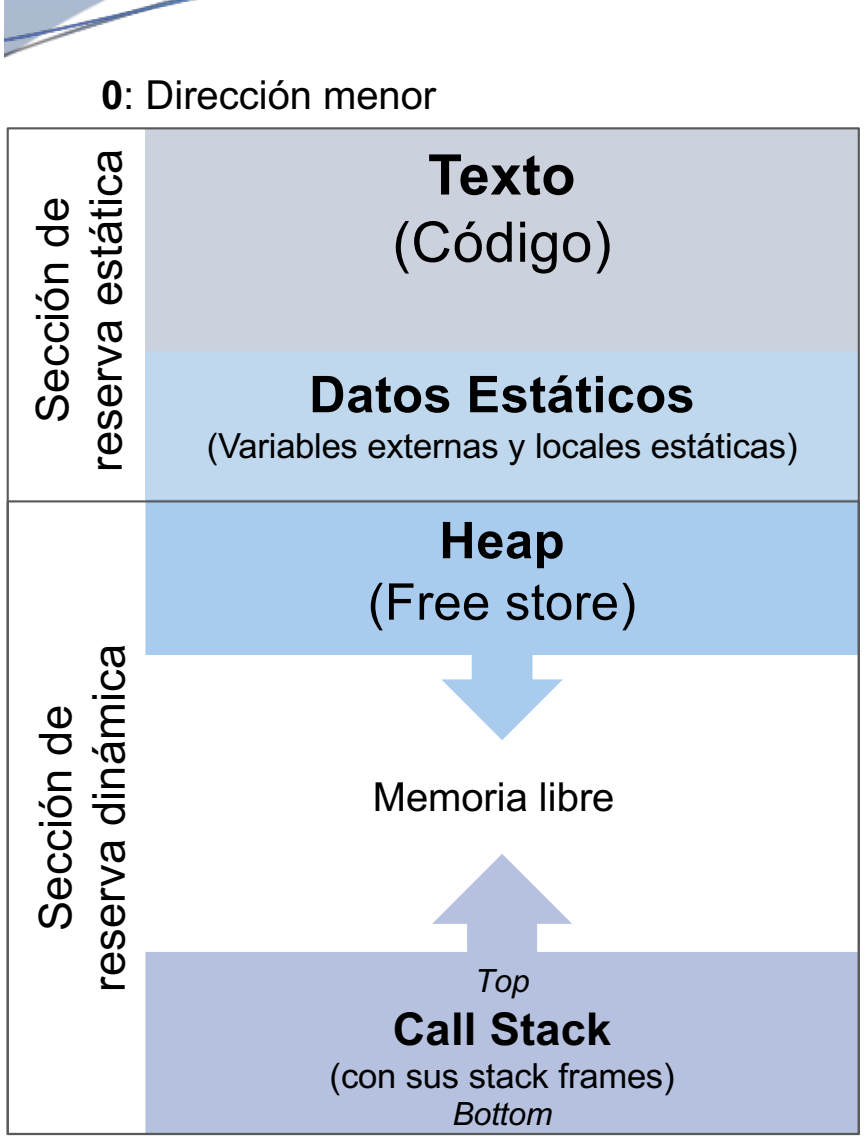
Layout de la Memoria Principal y Punteros

Layout (Disposición) de la Memoria




```
void f();
int* g(int);

static int a;       // Static

int main(){
  int b{1};         // Stack (automatic)

  f();

  int* p{&b};       // Stack
  *p = 3;

  p = new int;      // Heap
  *p = 4;
  delete p;

  p = g(21);
  delete p;
}


void f(){
  int c{2};         // Stack
  extern int a;     // No definition, just declaratio
  ++a;
  static int d{a}; // Static
  ++d;
}

int* g(int i){      // Stack
  int* p{new int}; // Stack y Heap
  *p = i;
  return p;
}
```

Empujando los Límites

Stack

```
int main(){
   void Chrome();
   Chrome();
}

void Chrome(){
   int a[42*42*42*42];
   Chrome();
}
```

Heap

```
int main(){
   void Chrome();
   Chrome();
}

void Chrome(){
   for(;;)
      new int[42*42*42*42];
}
```

Preguntas de Repaso Sobre Administración de Memoria y Punteros

- Dinámico y Estático
  - ¿Existe la memoria estática?
  - ¿Existe la memoria dinámica?
  - En el contexto de la administración de la memoria, ¿a qué se refieren estático y dinámico?
  - ¿Qué diferencias hay entre una variable constante y una estática?
  - ¿Puede una variable constante ser dinámica?
  - ¿Qué diferencias hay entre la reserva reserva estática y la dinámica?
- Stack
  - ¿Qué es es el *stack overflow*?
  - ¿Es dinámico?
  - ¿Quién y como se reserva?
  - ¿Quién y cómo se libera?
  - ¿Se puede no liberar?
- Heap
  - ¿Quién y como se reserva?
  - ¿Quién y cómo se libera?
  - ¿Qué pasa si no se libera la reserva?
  - ¿Qué ocurre si se usa un espacio liberado?
  - ¿Cuándo nos quedamos sin más heap?
- Pregunta avanzada
  - ¿Qué relación hay entre *el* heap y *un* heap?
  - ¿Qué diferencia hay entre el *heap* y el *free store*? ¿Aplica solo a C++? ¿Aplica a otros lenguajes?