Definición de struct

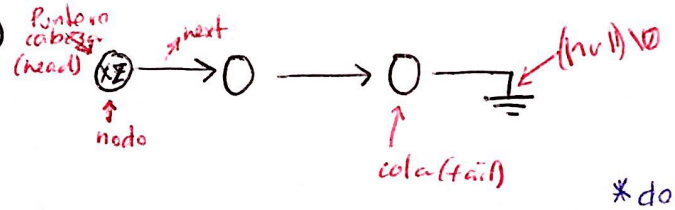
2019/08/05

```
struct nodo
{
  int y;
  int z;
  struct nodo *next;
}
```

define nodo a un puntero

Una lista simplemente encadenada

a)

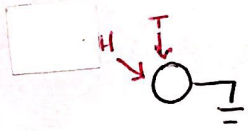


* un puntero tiene 8 bytes si es arquitectura de 64bits

* doblemente encadenada es que el último puntero es hacia adelante y hacia atrás
* encadenada circular: tail apunta al head

b) Creación de lista

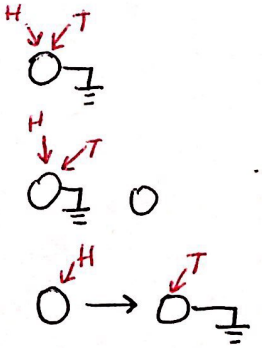
b1) Nueva lista



- Crear nodo (dibujar el círculo)
- nodo.next.null
- H ← nodo
  T ← nodo

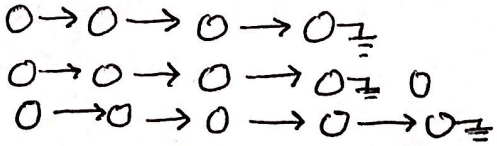
b2) Ya existe

b2.1)



- en términos de código b2.1 y b.2.2 es lo mismo
  - crear nodo
  - nodo.next ← null
  - tail.next ← nodo

b2.2) tiene más de un elemento



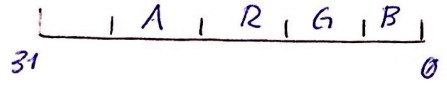
29/07/2019

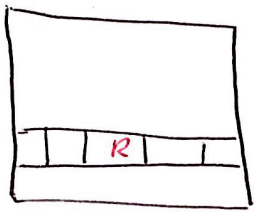
Las computadoras almacenan solo números en binario

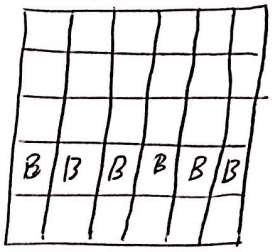
Image Codec:

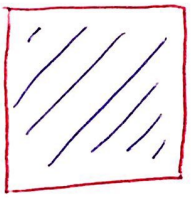
A R G B

Alpha Red Green Blue





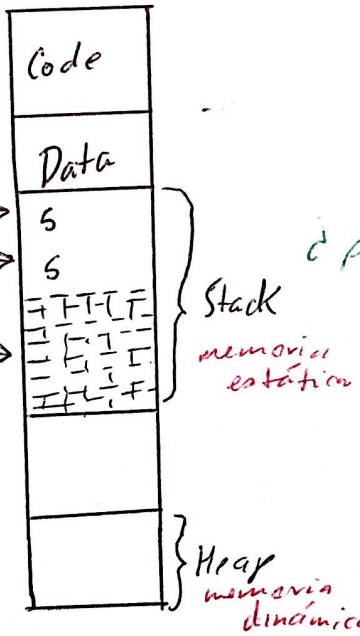




int a;

int *P;

int v[20];



malloc(size of memory)

↓ return

Puntero al espacio Reservado

↳ heap

¿Porqué en los ciclos se utilizan ijk y en los punteros pq?

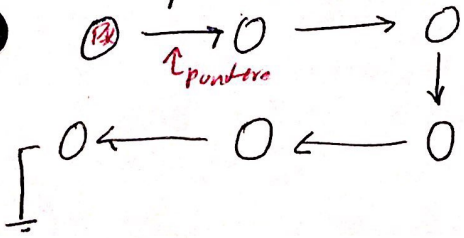
las strings se declaran como una lista indeterminada de elementos que para cuando llega a "null terminated"

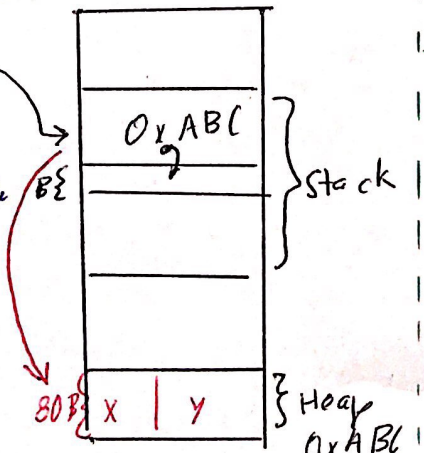
'hello \0'

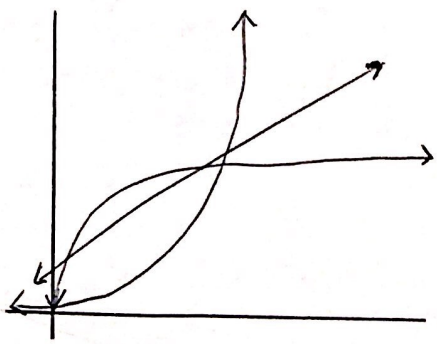
int *q;

q = malloc(80)

q = &B se pierde







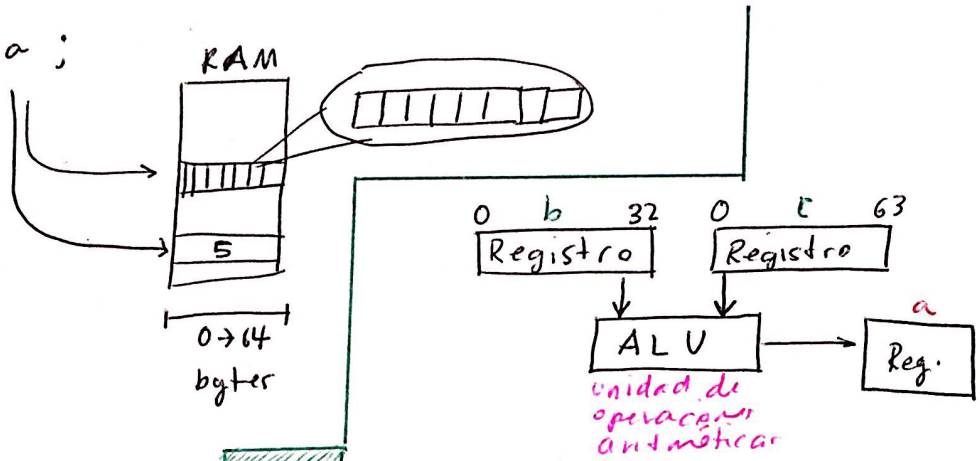
- el garbage collector libera el espacio en heap y las referencias en stack
- free(p)

22/07/2019

Lenguajes Interpretados, Compilados e Híbridos

en C... una variable se declara así

int a;

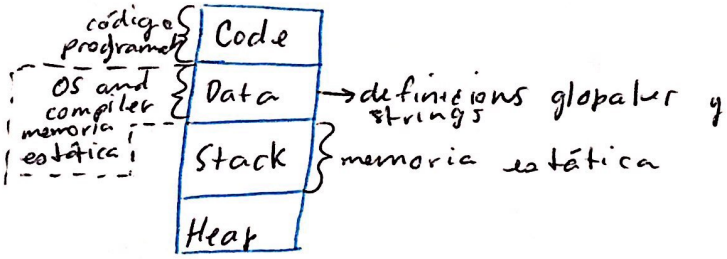


a: name,
Referencia,
Dirección

Memoria estática → desde la perspectiva de Procesos y O.S.

el apellido de la variable

int: tipo
↳ interpretación
↳ tamaño



Se inventó el puntero: es la estructura variable que almacena direcciones

Se ve cuando * antes de una variable.

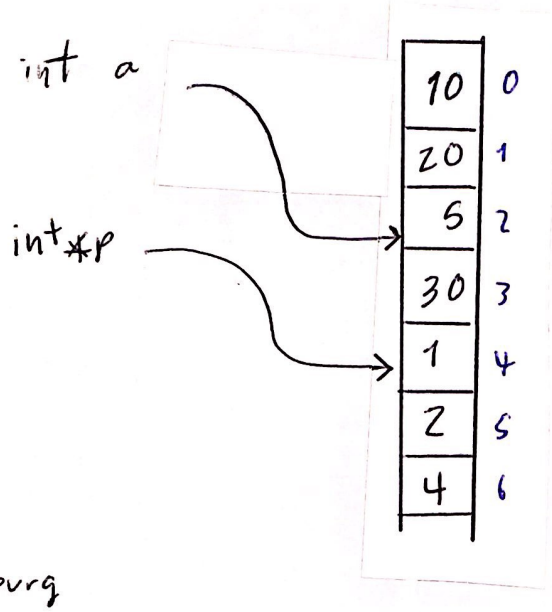
int *p | &a = dirección de a

*p obtiene el valor del espacio referenciado

- Con un puntero se accesa a los índices de memoria
- Programar con punteros es un arma de doble filo

Programación II

24/07/2019



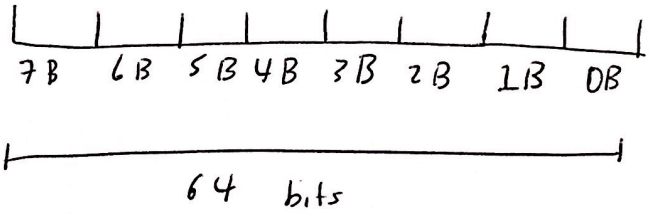
name	&name	*name
Print(a)	Print(&a)	Print(*a)
↓	↓	↓
5 (valor)	2 (dirección)	XXXX (indefinido)
Print(p)	Print(&p)	↓
↓	↓	20 (valor en la dir apuntada)
1 (valor)	4 (dir)	

Scaliburg

- No queremos salir del area segmentada

segmentos son bloques de memoria asignada para un proceso.

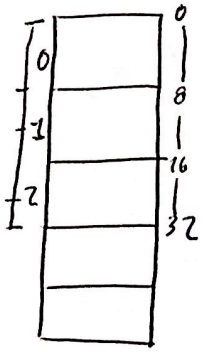
- las direcciones siempre van a bytes no a bits



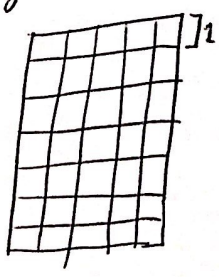
- Registro unidad mínima que utiliza para hacer el cálculo.

- Memoria

$$P_F = P_i + [index * size\ of(typ)]$$



- Todas las variables tienen word aligned



```
Struct  Estudiante
 {int   Carnet ;
  char  nombre [255];
  }
```