Punteros

Parte 0: Variables

Datos

Los datos de una aplicación se almacenan en la memoria , ésta consta de celdas numeradas de forma distintiva llamadas direcciones de memoria.

Cuando se almacena algo, es necesario conocer las direcciones para poder recuperarlo y trabajar con ello.

Variables

Un lenguaje de programación nos evita el tener que seguir el rastro de estas direcciones de memoria sustituyendo sus nombres.

Estos nombres se denominan variables. Las variables son los nombres descriptivos de las direcciones de memoria.

Declaración de variables

Nombre de la variable	Dirección de Memoria	Valor
	0x00C0	
	0x00C1	
	0x00C2	
	0x00C3	
	0x00C4	
	0x00C5	
	0x00C6	
	0x00C7	
	0x00C8	

Declaración de variables

```
int auxInt = 10;
```

Nombre de la variable	Dirección de Memoria	Valor
auxInt	0x00C0	10
	0x00C1	
	0x00C2	
	0x00C3	
	0x00C4	
	0x00C5	
	0x00C6	
	0x00C7	
	0x00C8	

Declaración de variables

```
int auxInt = 10;
```

```
printf("Dir: %p", &auxInt);
```

Nombre de la variable	Dirección de Memoria	Valor
auxInt	0x00C0	10
	0x00C1	
	0x00C2	
	0x00C3	
	0x00C4	
	0x00C5	
	0x00C6	
	0x00C7	
	0x00C8	

Declaración de variables

```
int auxInt = 10;
```

```
printf("Dir: %p", &auxInt);
```

>Dir: 0x00C0

Nombre de la variable	Dirección de Memoria	Valor
auxInt	0x00C0	10
	0x00C1	
	0x00C2	
	0x00C3	
	0x00C4	
	0x00C5	
	0x00C6	
	0x00C7	
	0x00C8	

Declaración de variables

```
int  auxInt = 10;
char    auxChar1 = ‘a’;
```

Nombre de la variable	Dirección de Memoria	Valor
auxInt	0x00C0	10
	0x00C1	
auxChar1	0x00C2	a
	0x00C3	
	0x00C4	
	0x00C5	
	0x00C6	
	0x00C7	
	0x00C8	

Declaración de variables

```
int  auxInt = 10;
char   auxChar1 = ‘a’;
char   auxChar2 = ‘B’;
```

Nombre de la variable	Dirección de Memoria	Valor
auxInt	0x00C0	10
	0x00C1	
auxChar1	0x00C2	a
auxChar2	0x00C3	B
	0x00C4	
	0x00C5	
	0x00C6	
	0x00C7	
	0x00C8	

Declaración de variables

```
int  auxInt = 10;
char    auxChar1 = ‘a’;
char    auxChar2 = ‘B’;
char    auxString[5] = “hola”;
```

Nombre de la variable	Dirección de Memoria	Valor
auxInt	0x00C0	10
	0x00C1	
auxChar1	0x00C2	a
auxChar2	0x00C3	B
auxString	0x00C4	‘h’
	0x00C5	‘o’
	0x00C6	‘l’
	0x00C7	‘a’
	0x00C8	‘\0’

Declaración de variables

```
int  auxInt = 10;
char    auxChar1 = ‘a’;
char    auxChar2 = ‘B’;
char    auxString[5] = “hola”;
```

auxInt = (int) &auxChar

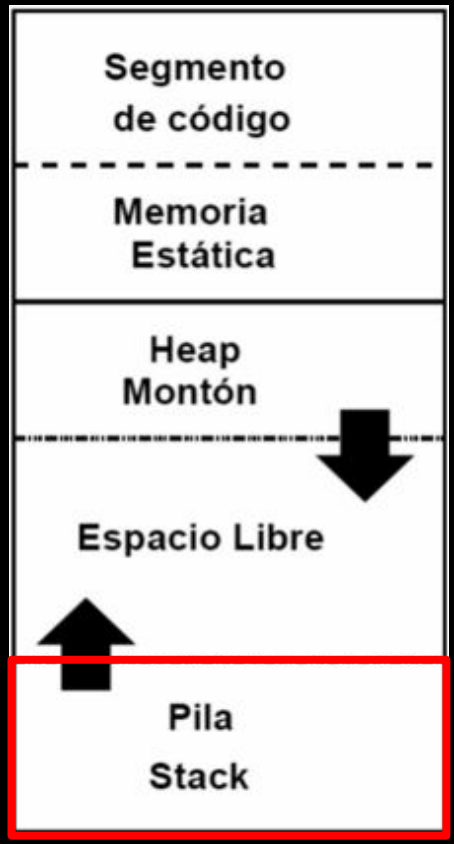
Nombre de la variable	Dirección de Memoria	Valor
auxInt	0x00C0	0x00C2
	0x00C1	
auxChar1	0x00C2	a
auxChar2	0x00C3	B
auxString	0x00C4	‘h’
	0x00C5	‘o’
	0x00C6	‘l’
	0x00C7	‘a’
	0x00C8	‘\0’

Stack

Segmento de Pila

Cada vez que se llama a una función entra en este segmento con toda su información y allí se guardan:

- Los llamados a las funciones
- Los parámetros de las funciones
- Las variables locales
- Otra información necesaria para el funcionamiento del programa.



Stack

```
int main (void ) {
    char a = '1' ;
    char b = '2' ;
    char c ;
    // swap
    c = a;
    a = b;
    b = c;
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
main	Variable	c	0x00C7	?
	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int main (void ) {
    char a = '1' ;
    char b = '2' ;
    char c ;
    // swap
    c = a;
    a = b;
    b = c;
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
main	Variable	c	0x00C7	1
	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int main (void ) {
    char a = '1' ;
    char b = '2' ;
    char c ;
    // swap
    c = a;
    a = b;
    b = c;
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
main	Variable	c	0x00C7	1
	Variable	b	0x00C8	2
	Variable	a	0x00C9	2
	Valor Ret.		0x00CA	

Stack

```
int main (void ) {
    char a = '1' ;
    char b = '2' ;
    char c ;
    // swap
    c = a;
    a = b;
    b = c;
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
main	Variable	c	0x00C7	1
	Variable	b	0x00C8	1
	Variable	a	0x00C9	2
	Valor Ret.		0x00CA	

Stack

```
int main (void ) {
    char a = '1' ;
    char b = '2' ;
    char c ;
    // swap
    c = a;
    a = b;
    b = c;
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
main	Variable	c	0x00C7	1
	Variable	b	0x00C8	1
	Variable	a	0x00C9	2
	Valor Ret.		0x00CA	0

Agreguemos la función swap()

Stack

```
int swap (char x , char y){
    char z ;
    z = x;
    x = y;
    y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(a , b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char x , char y){
    char z ;
    z = x;
    x = y;
    y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(a , b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char x , char y){
    char z ;
    z = x;
    x = y;
    y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(a , b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	
	Valor Ret.		0x00C2	
	Argumento	y	0x00C3	2
	Argumento	x	0x00C4	1
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char x , char y){
    char z ;
    z = x;
    x = y;
    y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(a , b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	1
	Valor Ret.		0x00C2	
	Argumento	y	0x00C3	2
	Argumento	x	0x00C4	1
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char x , char y){
    char z ;
    z = x;
    x = y;
    y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(a , b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	1
	Valor Ret.		0x00C2	
	Argumento	y	0x00C3	2
	Argumento	x	0x00C4	2
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char x , char y){
    char z ;
    z = x;
    x = y;
    y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(a , b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	1
	Valor Ret.		0x00C2	
	Argumento	y	0x00C3	1
	Argumento	x	0x00C4	2
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char x , char y){
    char z ;
    z = x;
    x = y;
    y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(a , b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	1
	Valor Ret.		0x00C2	0
	Argumento	y	0x00C3	1
	Argumento	x	0x00C4	2
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char x , char y){
      char z ;
      z = x;
      x = y;
      y = z;
      return 0;
}
int main (void ) {
      char a = '1', b = '2' ;
      swap(a , b);
      return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	0

¿Cómo lo solucionamos?

¿Cómo lo solucionamos?

Utilizando punteros

Stack

```
int swap (char* x , char* y){
    char z ;
    z = *x;
    *x = *y;
    *y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(&a , &b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char* x , char* y){
    char z ;
    z = *x;
    *x = *y;
    *y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(&a , &b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	?
	Valor Ret.		0x00C2	
	Argumento	y	0x00C3	0x00C8
	Argumento	x	0x00C4	0x00C9
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char* x , char* y){
    char z ;
    z = *x;
    *x = *y;
    *y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(&a , &b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	1
	Valor Ret.		0x00C2	
	Argumento	y	0x00C3	0x00C8
	Argumento	x	0x00C4	0x00C9
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	1
	Valor Ret.		0x00CA	

Stack

```
int swap (char* x , char* y){
    char z ;
    z = *x;
    *x = *y;
    *y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(&a , &b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	1
	Valor Ret.		0x00C2	
	Argumento	y	0x00C3	0x00C8
	Argumento	x	0x00C4	0x00C9
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	2
	Variable	a	0x00C9	2
	Valor Ret.		0x00CA	

Stack

```
int swap (char* x , char* y){
    char z ;
    z = *x;
    *x = *y;
    *y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(&a , &b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	1
	Valor Ret.		0x00C2	
	Argumento	y	0x00C3	0x00C8
	Argumento	x	0x00C4	0x00C9
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	1
	Variable	a	0x00C9	2
	Valor Ret.		0x00CA	

Stack

```
int swap (char* x , char* y){
    char z ;
    z = *x;
    *x = *y;
    *y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(&a , &b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
swap	Variable	z	0x00C1	1
	Valor Ret.		0x00C2	0
	Argumento	y	0x00C3	0x00C8
	Argumento	x	0x00C4	0x00C9
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	1
	Variable	a	0x00C9	2
	Valor Ret.		0x00CA	

Stack

```
int swap (char* x , char* y){
    char z ;
    z = *x;
    *x = *y;
    *y = z;
    return 0;
}
int main (void ) {
    char a = '1', b = '2' ;
    swap(&a , &b);
    return 0;
}
```

Funcion	Tipo	Nombre	Dirección	Valor
			0x00C0	
			0x00C1	
			0x00C2	
			0x00C3	
			0x00C4	
			0x00C5	
			0x00C6	
			0x00C7	
main	Variable	b	0x00C8	1
	Variable	a	0x00C9	2
	Valor Ret.		0x00CA	0