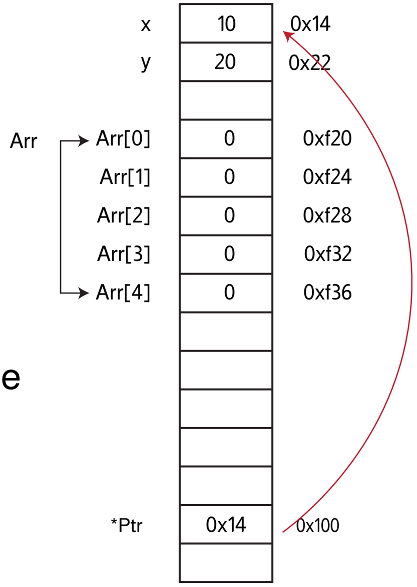
Day5_Cprogramming

Pointer
Pointer to pointer
Pointer to array
Struct

Data Types In C

- **int**
- **double**
- **float**
- **char**
- **void**
- **Array**
- **Pointer**
- **struct**

```
#include <stdio.h>


int main(){
      int x = 10;
      int y = 20;

      int arr[5] = {};
      sizeof(arr);//20Byte
printf(”%i”, x);
printf(”%i”, &x);

//Decliration
int *ptr;
//Assigment
ptr = &x;

int ptr2 = &x; //Compiler Error
return 0;
}
```



Name	Value	Address
x	10	0x14
y	20	0x22
arr	0xf20	0xf20
arr[0]	0	0xf20
ptr	0x14	0x100

&ptr --> Address Of Pointer It Self

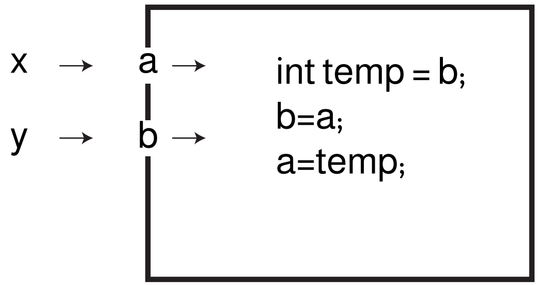
ptr --> Address Of Variable x

*ptr --> Value Of Address That Pointer Hold;

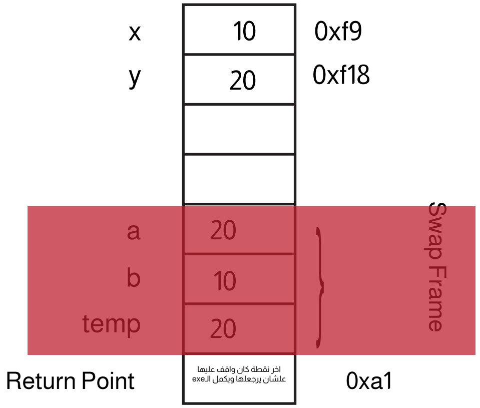
Pointer: Is Datatype That Can Hold address As Value
Pointer Size == 8Byte

المؤشرات: هى نوع من البيانات التى تحمل عنوان وليست قيمة اخرى

```
void Swap(int a, int b){
   int temp = b;
   b=a;
   a=temp;
}
```



```
int main()
{
      int x=10;
      int y = 20;
      printf("X Value = %i\n",x);
      printf("Y Value = %i\n",y);
→     Swap(x, y);
      printf("X Value = %i\n",x);
      printf("Y Value = %i\n",y);
return 0;
}
```



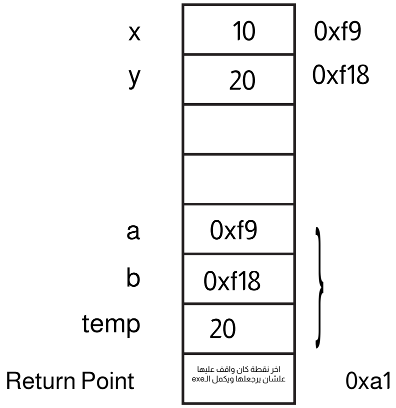
x	10	0xf9
y	20	0xf18
a	20	
b	10	
temp	20	
Return Point	اخر نقطة كان واقف عليها علشان يرجعلها ويكمل الـexe	0xa1

```
X Value = 10
Y Value = 20
X Value = 10
Y Value = 20
```

Pass By Values

```
void Swap(int *a, int *b){
   int temp = *b;
   *b=*a;
   *a = temp;
}
```

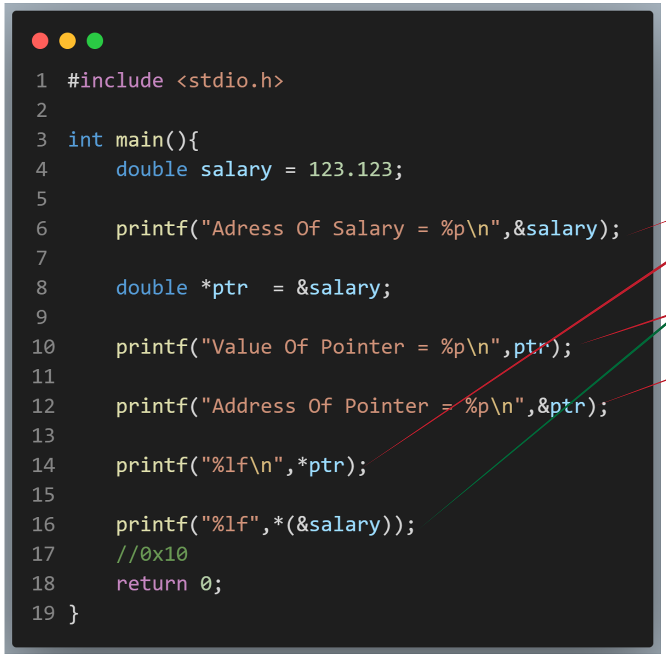
```
int main()
{
      int x=10;
      int y = 20;
      printf("X Value = %i\n",x);
      printf("Y Value = %i\n",y);
      Swap(&x, &y);
      printf("X Value = %i\n",x);
      printf("Y Value = %i\n",y);
return 0;
}
```



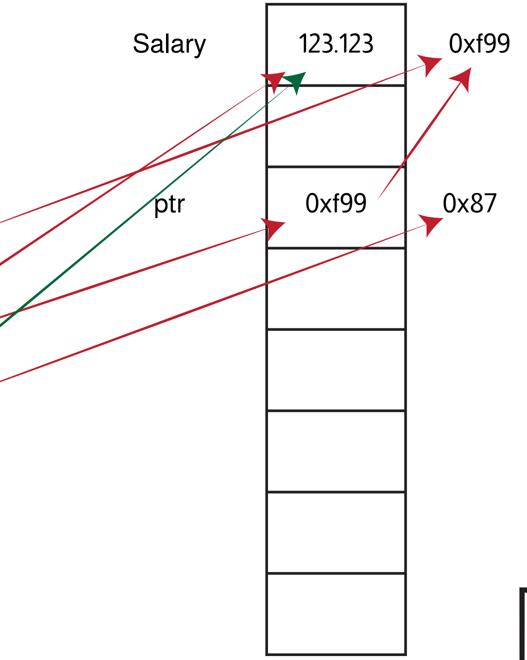
x	10	0xf9
y	20	0xf18
a	0xf9	
b	0xf18	
temp	20	
Return Point	اخر نقطة كان واقف عليها علشان يرجعلها ويكمل الـexe	0xa1

```
X Value = 10
Y Value = 20
X Value = 20
Y Value = 10
```

Pass By Ref

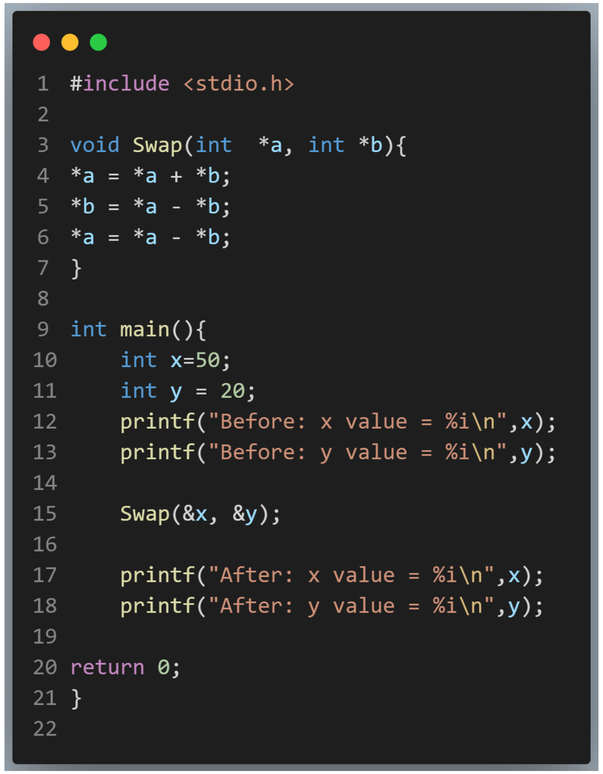


```
#include <stdio.h>

int main(){
    double salary = 123.123;

    printf("Adress Of Salary = %p\n",&salary);

    double *ptr  = &salary;

    printf("Value Of Pointer = %p\n",ptr);

    printf("Address Of Pointer = %p\n",&ptr);

    printf("%lf\n",*ptr);

    printf("%lf",*(&salary));
    //0x10
    return 0;
}
```

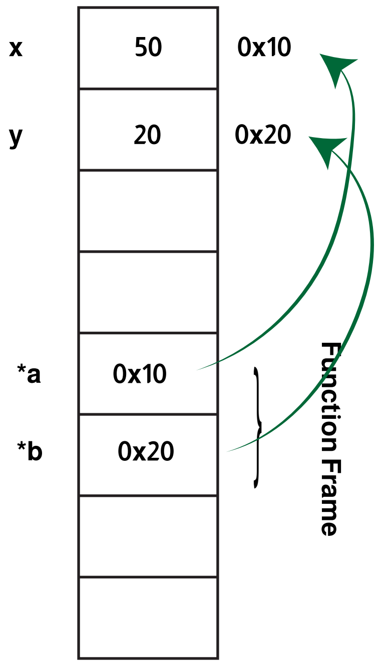


Name	Value
Salary	123.123
&Salary	0xf99
ptr	0xf99
&ptr	0x87
*(ptr)	123.123
*(&Salary)	123.123
*(0xf99)	123.123

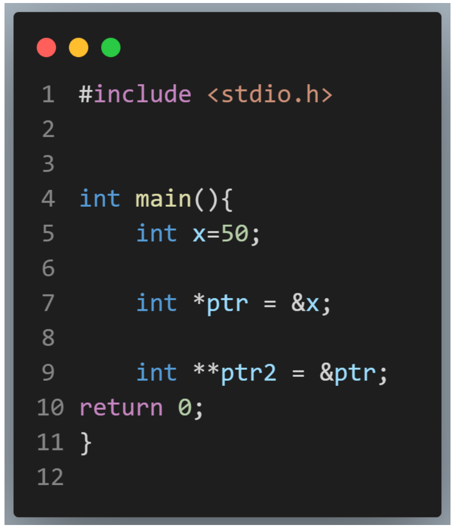
1. القيمة داخل المتغير Salary
2. عنوان المتغير Salary
3. القيمة بداخل المتغير Ptr
4. عنوان المتغير ptr
5. القيمة التي يحملها العنوان بداخل المتغير ptr
6. القيمة التي بداخل عنوان Salary
7. القيمة بداخل هذا العنوان



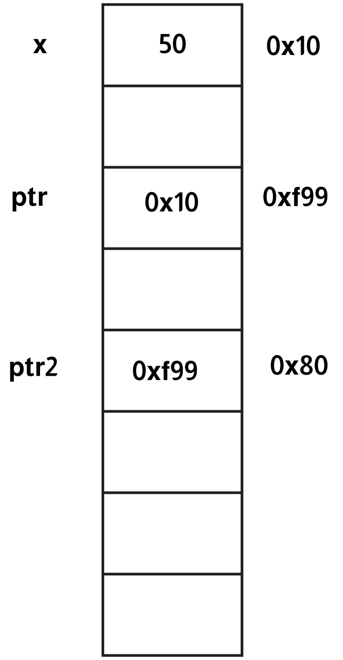
```
#include <stdio.h>

void Swap(int  *a, int *b){
*a = *a + *b;
*b = *a - *b;
*a = *a - *b;
}

int main(){
    int x=50;
    int y = 20;
    printf("Before: x value = %i\n",x);
    printf("Before: y value = %i\n",y);

    Swap(&x, &y);

    printf("After: x value = %i\n",x);
    printf("After: y value = %i\n",y);

return 0;
}
```



Pointer To Pointer



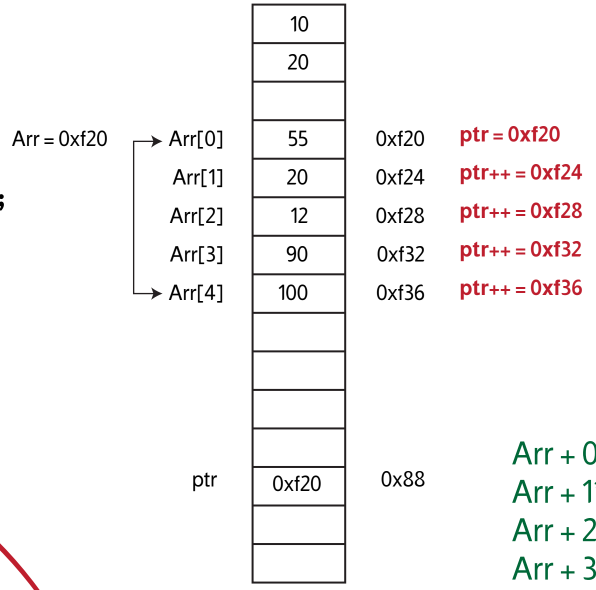
```
#include <stdio.h>


int main(){
    int x=50;

    int *ptr = &x;

    int **ptr2 = &ptr;
return 0;
}
```



1. ptr2 = 0xf99
2. &ptr2 = 0x80;
3. *ptr2 = 0x10
4. **ptr2 = 50
5. *(0x10) = 50

Pointer to Array

```
#include <stdio.h>


int main(){
    int Arr[5]= {55, 20, 12, 90, 100};

    int *ptr = &Arr[0];//true
    int *ptr = Arr;//true
    int *ptr = &Arr;//true
    Arr++;//Error

ptr++




return 0;
}
```



Arr = 0xf20

10
20

Arr[0] 55 0xf20 ptr = 0xf20
Arr[1] 20 0xf24 ptr++ = 0xf24
Arr[2] 12 0xf28 ptr++ = 0xf28
Arr[3] 90 0xf32 ptr++ = 0xf32
Arr[4] 100 0xf36 ptr++ = 0xf36

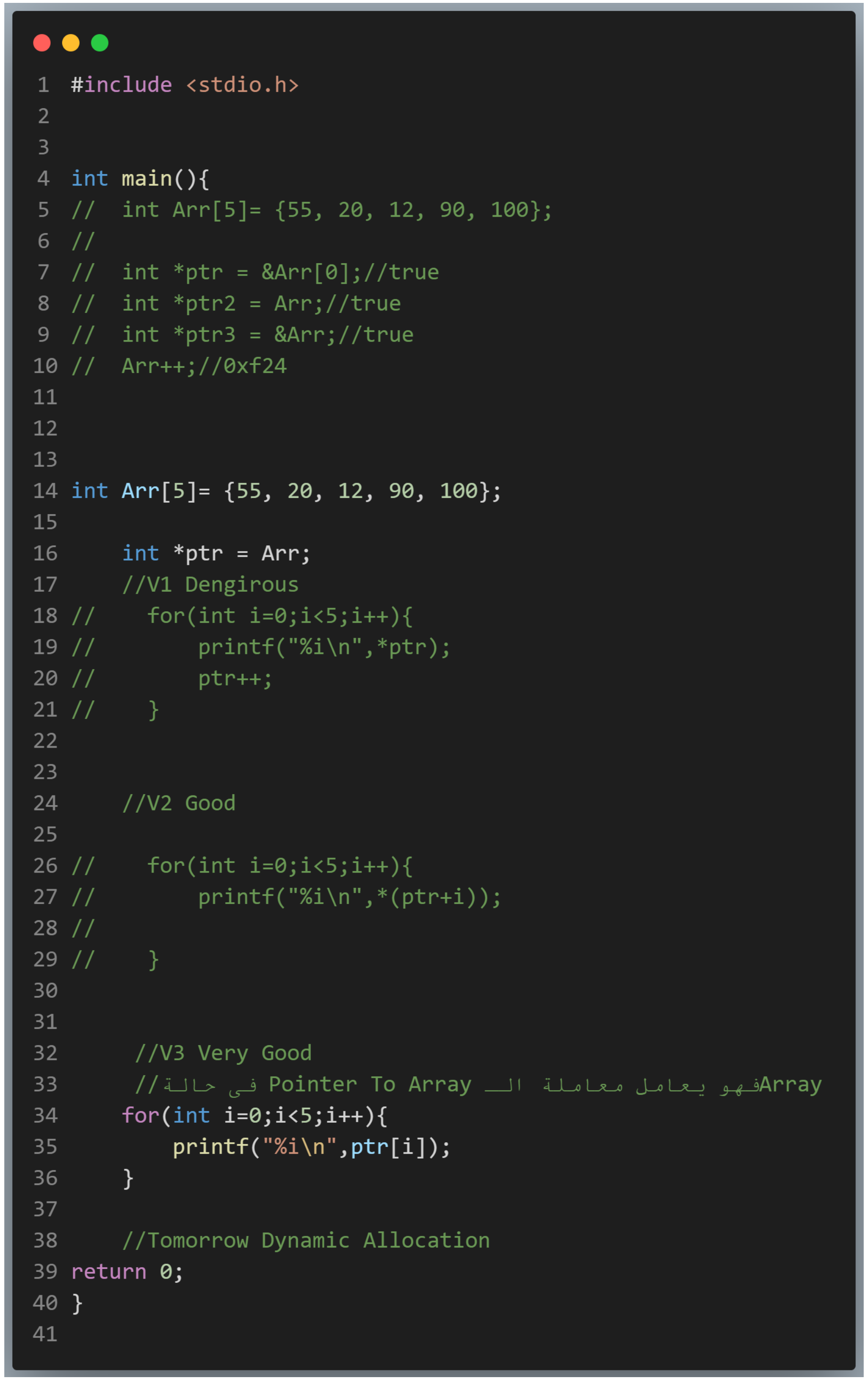
ptr 0xf20 0x88

Arr + 0*4 = 0xf20
Arr + 1*4 = 0xf24
Arr + 2*4 = 0xf28
Arr + 3*4 = 0xf32
Arr + 4*4 = 0xf36

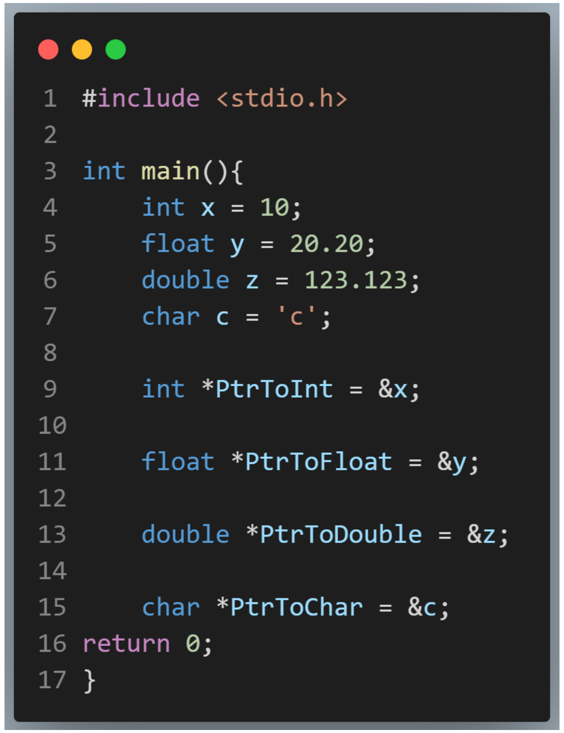
Name	Value	Address
Arr	0xf20	0x22
arr[0]	0	0xf20
ptr	0xf20	0x88

اسم الـArray يمثل عنوان أول Element ولا يمكن اى يتغير

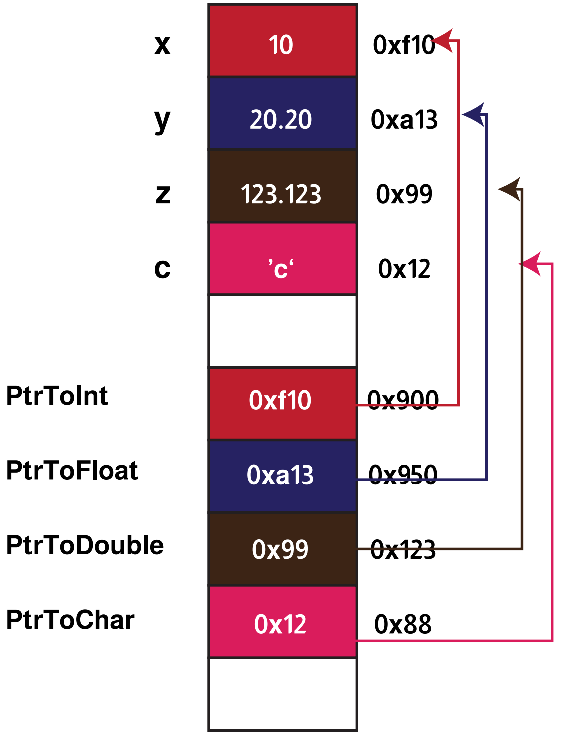
القاعدة بتقول فى حالة Pointer To Array يبقى الـPointer يعامل معاملة Array



```c
#include <stdio.h>


int main(){
//  int Arr[5]= {55, 20, 12, 90, 100};
//
//  int *ptr = &Arr[0];//true
//  int *ptr2 = Arr;//true
//  int *ptr3 = &Arr;//true
//  Arr++;//0xf24



int Arr[5]= {55, 20, 12, 90, 100};

    int *ptr = Arr;
    //V1 Dengirous
//    for(int i=0;i<5;i++){
//        printf("%i\n",*ptr);
//        ptr++;
//    }


    //V2 Good

//    for(int i=0;i<5;i++){
//        printf("%i\n",*(ptr+i));
//
//    }


     //V3 Very Good
     //فى حالة Pointer To Array فهو يعامل معاملة الـArray
    for(int i=0;i<5;i++){
        printf("%i\n",ptr[i]);
    }

    //Tomorrow Dynamic Allocation
return 0;
}

```



```c
#include <stdio.h>

int main(){
    int x = 10;
    float y = 20.20;
    double z = 123.123;
    char c = 'c';

    int *PtrToInt = &x;

    float *PtrToFloat = &y;

    double *PtrToDouble = &z;

    char *PtrToChar = &c;
return 0;
}
```



PtrToInt	-->	0xf10
PtrToFloat	-->	0xa13
PtrToDouble	-->	0x99
PtrToChar	-->	0x12
&PtrToInt	-->	0x900
&PtrToFloat	-->	0x950
&PtrToDouble	-->	0x123
&PtrToChar	-->	0x88
*PtrToInt	-->	10
*PtrToFloat	-->	20.20
*PtrToDouble	-->	123.123
*PtrToChar	-->	‘c’

Next Session

1. Struct
2. Dynamic Allocation