

Core Softskills Esstinall

22 Module

1. Os → Exam
2. Linux → Exam
3. Windows → Exam
4. Intro To Programming Using c → Exam
5. OOP Uisng cpp → Exam
6. Datastructre → Exam
7. Database using sql server → Exam
8. Network
9. Cloud
10. Tesing
11. Flow Charts
12. HTML → Exam
13. CSS → Exam
14. JS → Exam
15. Responsive Web
16. Wordpress
17. PHP
18. MySQL
19. Cummincation SKILLS
20. Interviewing Skills
21. CV Writing
22. Free lnace
23. Gen AI
- A. Project → Bussines Model

Day7_Cprogramming

Struct
Dynamic Allocation
Memory Type
Type Casting

Primitive

char → 1Byte

int → 4Byte

float → 4Byte

Double → 8Byte

Pointer → 8Byte

Non-Primitive

Array

Struct

User Define Data

```
#include <stdio.h>
#include <stdlib.h>
```

```
int main()
{
    float id;
    int age;
    char Name[20];

    float id2;
    int age2;
    char Name[20]2;

    float id3;
    int age3;
    char Name[20];
```

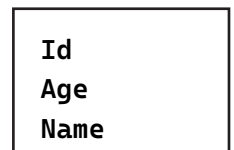
```
//Array
```

```
    return 0;
}
```

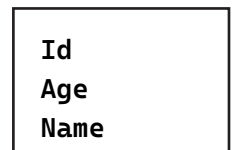
RAM
Random Access Memory



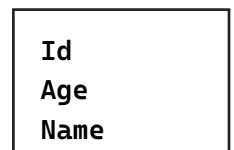
Key



Key



Key



Struct → User Define Datatype, Can Carry Multy Variable From Def Datatype

```
#include <stdio.h>
#include <stdlib.h>
```

```
struct Employee{ //Logical Representation
    float Id; → 4
    int Age; → 4
    char Name[20]; → 20Byte
}
```

```
int main()
{
    int x = 10;
    //Decliration
    struct Employee emp1; //Physcial Reprisintation
    emp1.id = 123;
    emp1.age = 28;
    //emp1.Name="Mustafa"
    emp1.Name[0]='M';
    strcpy(emp1.Name,"Mustafa");
    return 0;
}
```

□

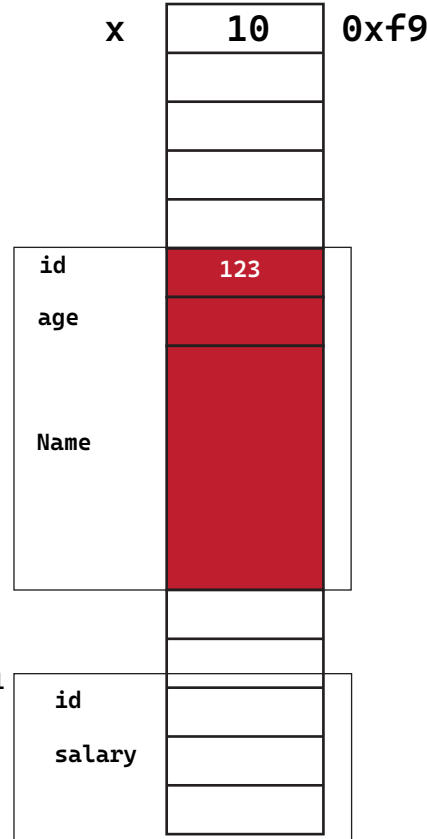
Local Variable

x

كل المتغيرات المعرفة داخل main

emp1

s1



Member Variable

emp1

كل المتغيرات المعرفة داخل Struct

struct DataTypeName{

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 //Logical Representation
5 //Blueprint
6 struct Employee{
7     float id;
8     int age;
9     char Name[20];
10 };
11
12 int main()
13 {
14     //Decliration
15     struct Employee e1;
16     //Assignment
17     e1.id = 123;
18     e1.age = 28;
19     strcpy(e1.Name,"Mustafa");
20     //Call
21     printf("Employee Id = %f\n",e1.id);
22     printf("Employee Age = %i\n",e1.age);
23     printf("Employee Name = %s\n",e1.Name);
24
25     struct Employee e2;
26     e2.id=456;
27     strcpy(e2.Name,"Ahmed");
28     e2.age=30;
29     printf("Employee 2 Id = %.0f\n",e2.id);
30     printf("Employee 2 Age = %i\n",e2.age);
31     printf("Employee 2 Name = %s\n",e2.Name);
32     return 0;
33 }
34
```

```
#include <stdio.h>
#include <stdlib.h>
```

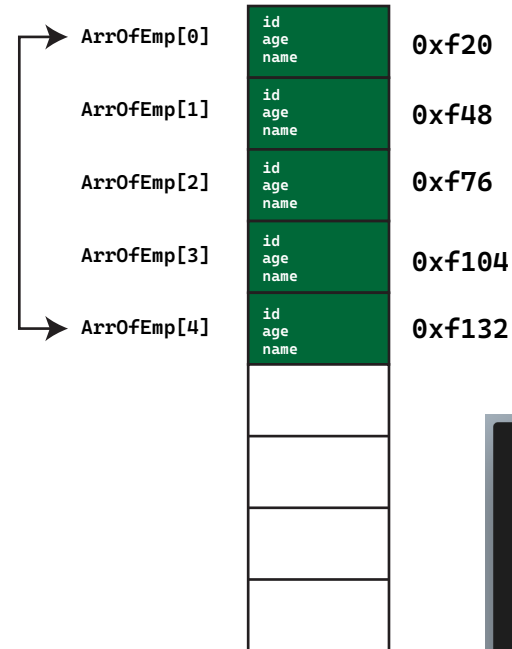
```
struct Employee{ //28Byte
    int id;
    int age;
    char Name[20];
};
```

```
int main()
{
    struct Employee ArrOfEmp[5]; //280
    for(int i = 0; i < 5; i++){
        printf("Please Enter Employee Number %i Id", i + 1);
        scanf("%i", &ArrOfEmp[i].id);
        printf("Please Enter Employee Number %i Age", i + 1);
        scanf("%i", &ArrOfEmp[i].age);
        printf("Please Enter Employee Number %i Name", i + 1);
        scanf("%s", &ArrOfEmp[i].Name);
    }

    for(int i=0; i<5; i++){
        printf("Employee Number %i Id: %i", i+1, ArrOfEmp[i].id);
        printf("Employee Number %i Age: %i", i+1, ArrOfEmp[i].age);
        printf("Employee Number %i Name: %s", i+1, ArrOfEmp[i].Name);
    }

    return 0;
}
```

ArrOfEmp = 0xf20



```
ArrOfEmp[0].idv
ArrOfEmp[0].age
ArrOfEmp[0].name
ArrOfEmp[1].id
ArrOfEmp[1].age
ArrOfEmp[1].name
ArrOfEmp[2].id
ArrOfEmp[2].age
ArrOfEmp[2].name
ArrOfEmp[3].id
ArrOfEmp[3].age
ArrOfEmp[3].name
ArrOfEmp[4].id
ArrOfEmp[4].age
ArrOfEmp[5].name
```

Array Of Struct

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 struct Employee{ //28Byte
4     int id;
5     int age;
6     char Name[20];
7 };
8 int main()
9 {
10     struct Employee ArrOfEmp[5]; //280
11     for(int i = 0; i < 5; i++){
12         printf("Please Enter Employee Number %i Id\n", i + 1);
13         scanf("%i", &ArrOfEmp[i].id);
14         printf("Please Enter Employee Number %i Age\n", i + 1);
15         scanf("%i", &ArrOfEmp[i].age);
16         printf("Please Enter Employee Number %i Name\n", i + 1);
17         scanf("%s", &ArrOfEmp[i].Name);
18     }
19
20     for(int i=0; i<5; i++){
21         printf("Employee Number %i Id: %i\n", i+1, ArrOfEmp[i].id);
22         printf("Employee Number %i Age: %i\n", i+1, ArrOfEmp[i].age);
23         printf("Employee Number %i Name: %s\n", i+1, ArrOfEmp[i].Name);
24         printf("\n\n_____ \n\n");
25     }
26
27     return 0;
28 }
29
```

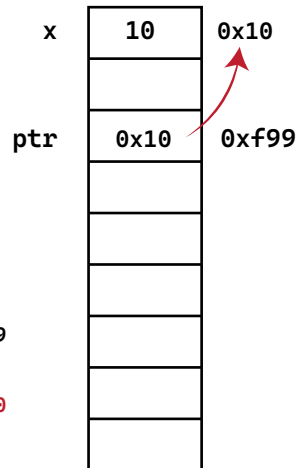
Pointer To Struct

```
#include <stdio.h>
#include <stdlib.h>
```

```
int main()
{
    int x = 10;
    int *ptr = &x;

    printf("%p\n", ptr); //0xf99
    printf("%p\n", &ptr); //0xf99
    printf("%i\n", *ptr); //10
    printf("%i\n", *(0x10)); //10

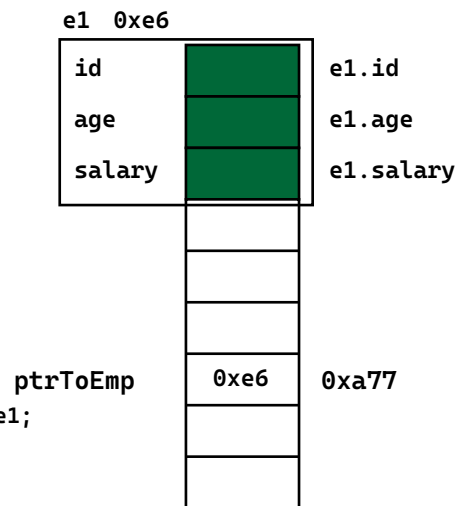
    return 0;
}
```



```
#include <stdio.h>
#include <stdlib.h>

struct Employee{ //12Byte
    int id;
    int age;
    float salary;
}
```

```
int main()
{
    struct Employee e1;
    struct Employee *ptrToEmp = &e1;
    ptrToEmp //0xe6
    (*ptrToEmp).id
    ptrToEmp->id;
    ptrToEmp->age;
    ptrToEmp->salary;
    return 0;
}
```



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 struct Employee{ //28Byte
4     int id;
5     int age;
6     float salary;
7 };
```

```
ptrToEmp->id;
ptrToEmp->age;
ptrToEmp->salary;
```

Name	value	Address
x	10	0x10
ptr	0x10	0xf99
e1	0xe6	0xe6
ptrToEmp	0xe6	0xa77

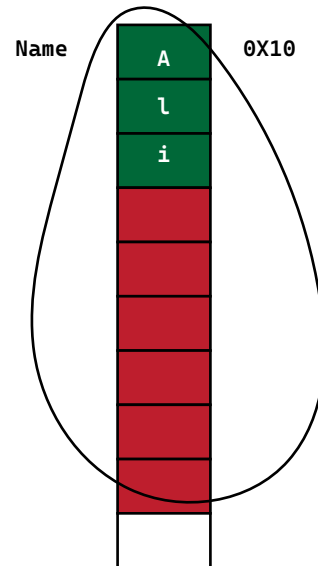
```
1 int main()
2 {
3     struct Employee e1;
4     // e1.id=10;
5     // e1.age=28;
6     // e1.salary=5000;
7     // printf("Id: %i\n", e1.id);
8     // printf("Age: %i\n", e1.age);
9     // printf("Salary: %f\n", e1.salary);
10    struct Employee *ptrToEmp = &e1;
11
12    //(*ptrToEmp).id=123;
13    //printf("Id: %i\n", (*ptrToEmp).id);
14
15    ptrToEmp->id = 123;
16    ptrToEmp->age=28;
17    ptrToEmp->salary=6000;
18
19    printf("Id: %i\n", ptrToEmp->id);
20    printf("Age: %i\n", ptrToEmp->age);
21    printf("Salary: %f\n", ptrToEmp->salary);
22
23    return 0;
24 }
```

Dynamic Allocation

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    char Name[100];
    printf("Please Enter Your Name\n");
    fgets(Name,sizeof(Name),stdin);

    printf("Your Name Is %s",Name);
    return 0;
}
```



OS → RAM

os Has Full Control To Manage Ram
الـ os هو المسؤول عن تحديد المساحة المحجوزة في الذاكرة

Array is A Compile Time Data Type

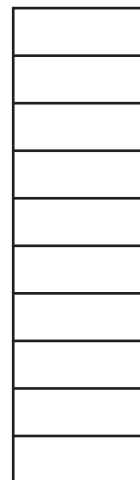
معناها ان لازم وقت الـ Compile يكون OS عارف هيروح يحجز مساحة كام BYTE للمصفوفة

الحل:

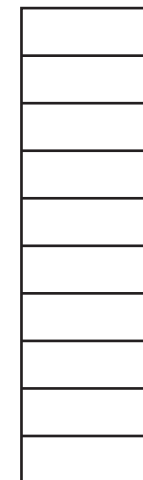
استخدام Dynamic Allocation

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 int main()
6 {
7     int size;
8     printf("Please Enter Name Number Of Character\n");
9     scanf("%i",&size);
10    _flushall();
11
12    char Name[size];
13    printf("Please Enter Your Name\n");
14    fgets(Name,sizeof(Name),stdin);
15
16
17    printf("Your Name Is %s",Name);
18    return 0;
19 }
20
```

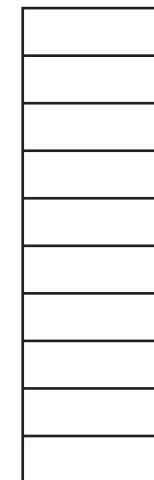
Stack
OS



Heap



BSS

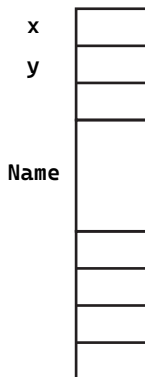


Stack

os هو المسؤول عن حجز المساحات في Stack وقت Compile Time

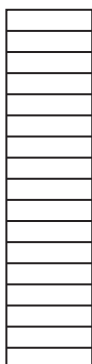
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
int main()
{
    int x = 10;
    int y = 20;
    char Name[30];
    return 0;
}
```

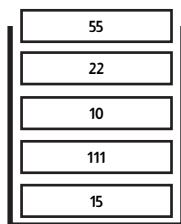


Stack
LIFO
Last in First Out

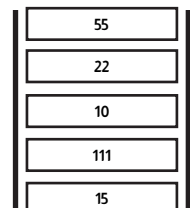
```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 int ReturnValue(){
5     int value;
6 }
7 void Swap(int a, int b){
8     int val = ReturnValue();
9 }
10 int main()
11 {
12     int x;
13     int y;
14     swap(x, y);
15     return 0;
16 }
17
```



Value
val
b
a
y
x



Queue
FIFO
First In First Out



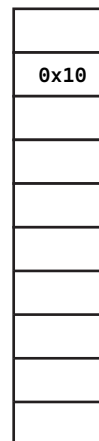
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
int main()
{
    int *Ptr;
    Ptr = malloc(20);

    return 0;
}
```

Stack

Ptr



0xe8

Heap

User يتمكن من حجز المساحات المستخدمة بنفسه، وقت Run Time



Casting

هو عملية تحويل القيم من نوع بيانات لنوع اخر

Implicit

الكومبايلر هو الى يقوم بعملية الكاستنج

Explicit

Dynamic Allocation

يتسمح للمستخدم وقت Run Time بحجز وتعديل المساحة المطلوبة
Life time بتاعها غير متوقف على تحرير المستخدم لهذه المساحة

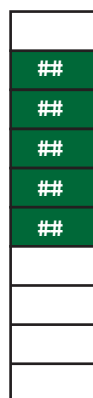
void malloc(20)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
int main()
{
    int Arr[5];
    return 0;
}
```

Stack

Arr



Stack

Ptr



Heap

0xf10

00000000 00000000 00010000 01000000



```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
int main()
{
    int size = 12
    int *Ptr;
    Ptr = malloc(size);
    free(Ptr);
    return 0;
}
```

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 int main()
6 {
7     int *Ptr;
8     // printf("%i", sizeof(int));
9     int size;
10    printf("Please Enter Size Of Array\n");
11    scanf("%i", &size);
12    Ptr = malloc(sizeof(int) * size);
13
14    for(int i=0; i<size; i++){
15        printf("Please Enter Value Number %i\n", i+1);
16        scanf("%i", &Ptr[i]);
17    }
18
19    for(int i=0; i<size; i++){
20        printf("%i\n", Ptr[i]);
21        printf("%p\n", &Ptr[i]);
22    }
23
24 }
25
26 return 0;
27 }
28

```

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 struct Employee{
4     int id;//4
5     char Name[20]; //20
6     float Salary; //28
7 };
8 int main()
9 {
10     int size;
11     printf("Please Enter Number Of Employee\n");
12     scanf("%i", &size);
13
14     struct Employee *Ptr;
15     Ptr = malloc(sizeof(struct Employee) * 1);
16
17     for(int i=0; i<size; i++){
18         printf("Please Enter Employee Number %i, Id\n", i+1);
19         scanf("%i", &Ptr[i].id);
20         printf("Please Enter Employee Number %i, Salary\n", i+1);
21         scanf("%f", &Ptr[i].Salary);
22         printf("Please Enter Employee Number %i, Name\n", i+1);
23         scanf("%s", &Ptr[i].Name);
24     }
25 //Print All Data
26 for(int i=0; i<size; i++){
27     printf("Employee Number %i, Id: %i\n", i+1, Ptr[i].id);
28     printf("Employee Number %i, Name: %s\n", i+1, Ptr[i].Name);
29     printf("Employee Number %i, Salary: %.0f\n", i+1, Ptr[i].Salary);
30 }
31

```

```

1
2 //Search
3 printf("\n\n\n_____ \n\n\n");
4 int Id;
5 printf("Please Enter Id Value To Seach\n");
6 scanf("%i", &Id);
7 for(int i=0; i<size; i++){
8     if(Ptr[i].id == Id){
9         printf("Employee Number %i, Id: %i\n", i+1, Ptr[i].id);
10        printf("Employee Number %i, Name: %s\n", i+1, Ptr[i].Name);
11        printf("Employee Number %i, Salary: %.0f\n", i+1, Ptr[i].Salary);
12        break;
13    }
14 }
15
16 //Edit
17 printf("\n\n\n_____ \n\n\n");
18 Id;
19 printf("Please Enter Id Value To Edit\n");
20 scanf("%i", &Id);
21 for(int i=0; i<size; i++){
22     if(Ptr[i].id == Id){
23         printf("Please Enter Employee Number %i, Id\n", i+1);
24         scanf("%i", &Ptr[i].id);
25         printf("Please Enter Employee Number %i, Salary\n", i+1);
26         scanf("%f", &Ptr[i].Salary);
27         printf("Please Enter Employee Number %i, Name\n", i+1);
28         scanf("%s", &Ptr[i].Name);
29         break;
30     }
31 }
32
33 printf("\n\n\n_____ \n\n\n");
34 for(int i=0; i<size; i++){
35     printf("Employee Number %i, Id: %i\n", i+1, Ptr[i].id);
36     printf("Employee Number %i, Name: %s\n", i+1, Ptr[i].Name);
37     printf("Employee Number %i, Salary: %.0f\n", i+1, Ptr[i].Salary);
38 }
39
40
41 free(Ptr);
42
43
44
45 // Try All What We Learn In Lecture
46 return 0;
47 }
48

```

C:\Users\yoppa\Desktop\abv.exe

Please Select From The Menu

- 1- Read Employees
- 2- Add New Employee
- 3- Edit Employee
- 4- Delete Employee

Your Choice Is:

Reference

Final C Project