

## الدرس السابع

النهاردة هانشرح حاجة كتير من الناس عندهم مشكلة فيها

هانشرح بوينتر وماأدراك مالوبينتر

عايزين تركيز شوية بس

نبدأ ... مستعدين

pointer

لما بعرف فاربيال مش بيروح في الميموري وبيتخزن فيها

طب اكيد بيقاله عنوان

address

عشان الكومبيلر

compiler

يعرف هي فين ويروحها كل ماستخدمها

طيب انا لما بعرف فاربيال

int x=0;

مثلا ... انا عرفت ان فيه 0 بس مش عارف العنوان بتاعه

لو عايز اجيب العنوان اعمل ايه؟؟

في حاجة بتجيب العنوان بتاع الفاربيال وهي

& reference(address)

دي احطها قبل الفاربيال هاتجيلي العنوان بتاعه

يعني لو عملت

cout<<&x;

x هاتجيلي العنوان بتاع الفاربيال

في الميموري

طيب ينفع اعمل كدة؟؟

int y;

y=&x;

cout<<y;

زي اللي فوق بس بدل ماعملت سي اوت علي طول

حطتها في فاربيال وعملتله سي اوت

ينفع؟؟ فكر شوية كدة

لا ... طب لا ليه

عشان انا حطيت عنوان في فاربيال من النوع انتجر

والعنوان ده مش انتجر .. تمام

طب ايه الحل

قالك خلاص هانعرف حاجة اسمها بوينتر

Pointer

يبقي بيشيل جواه عنوان زي مالانتجر بيشيل انتجر

طيب هافرق ازاي ان ده بوينتر

مش لما كنت بعرف الفاربيال كان بيبقي كدة

int x;

ازاي بقي اعرف البوينتر وافرق انه بوينتر مش فاربيال عادي

\*asteriskخلاص نخط قدامها استريسك

عشان اعرف ان ده بوينتر

int \*x;

ايه ده ...؟؟

في حاجة غريبة مش كدة

هو لسة من النوع انتجر

طب ازاي؟؟

مش قولت ده بوينتر ازاي بقي بيبقي من النوع انتجر

لا بص ... انا هنا بقوله ان هو بوينتر يعني بيشيل عنوان

x\*ودي اللي هي

انما بقوله هو بيشيل عنوان لفاربيال من النوع انتجر

intودي هي

يعني انا بقله هو بوينتر بيشاور علي عنوان فاربيال ن النوع انتجر

```
int *x
```

يعني البوينتر يعتبر فاربيال عادي مش نوع

مش ( data type )

بس القيمة اللي جواه هي عنوان فاربيال ثاني

وهي باينة من اسمها

Pointer

point to address يعني

يعني بيشاور علي عنوان

طيب نعمل برنامج صغير لبوينتر بيشاور علي انتجر ونأكد من انه فعلا

بيشاور علي العنوان الصح

```
int main()
```

```
{
```

```
int x=2 , *p=0;
```

```
cout<<&x<<endl;
```

```
p=&x;
```

```
cout<<p<<endl;
```

```
cout<<*p<<endl;
```

```
return 0;
```

```
}
```

البرنامج ده هيعمل ايه ... تعالى نشوف

في الاول خالص انا عرفت فاربيال انتجر

```
int x=2;
```

وبعدين عرفت بوينتر من النوع انتجر برده

وده معناه انه بيشاور علي انتجر

تمام

بعد كدة قولتله

```
cout<<&x<<endl;
```

هنا هاخرج العنوان بتاع الفاريبال زي ماتفقنا

```
p=&x;
```

هنا هايجيب العنوان بتاع الفاريبال ويحطه في البوينتر

تمام

ينفع كدة ؟؟

ايوة تمام ينفع عشان الاتنين عناوين فينفع

بعد كدة

```
cout<<p<<endl;
```

قولنا ده بوينتر بيقى بيثيل عنوان

فهايطلع العنوان الي بيشار عليه وهو

x

يبقى هايطلع العنوان اللي طلعه قبل كدة

لما عملت

```
cout<<&x<<endl;
```

بعد كدة

```
cout<<*p<<endl;
```

هنا بقي بقله طلع القيمة اللي انت بتشار عليها

فهايطلع 2

تمام كدة

دي فكرة البوينتر انه بيثيل عنوان لفاريبال ثاني

ممكّن يكون من اي نوع

طيب فاكرين الاراي

طبعا فاكرنها .. فيها مشكلة

اني لازم احدد الحجم بتاعها قبل مابدأ البرنامج

ودي مشكلة مش كدة ... طيب ايه الحل

الحل اني اعمل

dynamic array

يعني اخلي الحجم بتاعها مش ثابت

طيب نعمل كدة ازاي

ده هابتعمل بالوينتر

مممممممم .. ايه اللي انت بتقوله ده

وايه اصلا العلاقة بين الاراي والوينتر

طيب الاول فعلا لازم نعرف العلاقة بين الاراي والوينتر عشان اعرف ازاي هاعمل ارأي مش ثابت الحجم بيها

ركز شوية بقي معاليا

بص الاراي يعتبر بوينتر بس ثابت

ازاي؟؟

الاراي مش بقول انها عبارة عن مجموعة قيم من نفس النوع

يعني لما بعمل مثلا ارأي ل10 قيم

```
int arr[10];
```

قولنا انه بيحجز في الميموري 10 اماكن

تمام

طيب مايتعبّر اول

element

في الاراي عامل زي البوينتر لانه بيوصلني لبقية عناوين عناصر الاراي

مش كدة .. عشان كدة يعتبر الاراي بوينتر بس ثابت

```
constant pointer = array
```

يعني ينفع اعمل كدة

```
int arr[10] , *p;
```

```
p=arr;
```

ايوة ينفع عشان اتفقنا ان الاراي عبارة عن بوينتر بس ثابت

طيب سؤال بقي للناس وياريت تفكروا فيه

ينفع اعمل كدة

```
p=arr;
```

بس مينفعش اعمل كدة

```
arr=p;
```

ليه ؟؟

هاسييك الاجابة وممكن تبعتلي وأقولك صح ولا غلط لو مهتم تعرف الاجابة يعني

طيب عايزين نعمل اراي متغير الحجم بالوينتر بقي

يعني هاشتغل علي

dynamic array

اول ماعمل حاجة في السي بلس بلس متغيرة الحجم

يعني dynamic

يبقي هاستخدم حاجتين

new , delete

لما ابدأ اعمل حاجة متغيرة عشان اعرفه new

dynamic انها هاتبقي

اني هاحجز مكان معين للبرنامج بتاعي بس

delete

هنا بقي هاستخدمها لما اخلص عشان اخلي الميموري اللي خصصتها

للبرنامج بتاعي تبقي متاحة بقي لاي برنامج ثاني

تمام كدة ؟؟

هاختم بقي بكود اراي متغير الحجم

```
#include <iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
int x=0;
```

```
    cin>>x;
```

```
    int *arr;
```

```
    arr = new int [x];
```

```
    for(int i=0 ; i<x;i++)
```

```

cin>>arr[i];

for(int i=0 ; i<x;i++)

cout<< arr[i];

delete [] arr;

system("PAUSE");

return 0;

}

```

هركز علي الحاجات الغريبة في الكود

طبعا الاول عرفت فاربيال

```
int x=0;
```

وهاخذ من اليوزر قيمته اللي هيا حجم الاراي

وبعدين عرفت الاراي

```
int *arr;
```

```
arr = new int [x];
```

مش اتفنا اني هاعملها متغيرة الحجم

dynamic

فعرفت الاراي عن طريق البوينتر يعني عشان يبقى مش ثابت الحجم وقولت

انه هايحجز مكان في الميموري لمجموعة فاربيال من النوع انتجر

وعملت سي ان وسي اوت عادي للاراي

خدناها المرة اللي فاتت دي

وبعدين مسحت الميموري زي ماتفقنا عشان اخلي المساحة اللي حجزتها متاحة لاي برنامج ثاني

```
delete [] arr;
```

مكتوبة كدة ليه؟؟

بص

لو هاعمل بوينتر لحاجة واحدة تبقى

```
بسnew
```

```
int *p;
```

```
p=new int
```

انما لو هاتشاور علي كذا حاجة يبقي

```
p=new int [size]
```

والمسح كدة برده

```
delete
```

```
delete p;
```

لو واحدة

وهاتبقي

```
delete [] p;
```

لو كذا حاجة

بس كدة دي فكرة البوينتر باختصار

ومتنسوش في سؤال فوق للي مهتم ممكن بيعت اجابته او تفكيره وانا هارد عليه

اسئلة للي عايز يتابع الشرح

1- what is the use of :pointer , new , delete

2- what is the relation between pointer and array ?

**send solution or any question to**

**smile4ever\_1989@yahoo.com**

**Made by**

**Ibrahim Azmy**

**Student Union**