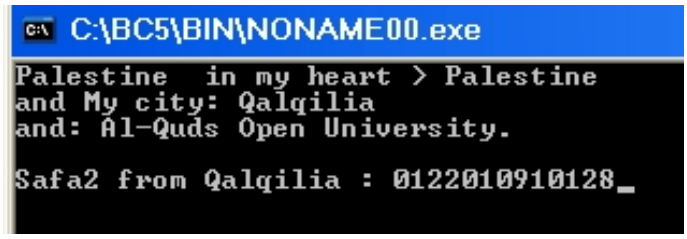
بسم الله الرحمن الرحيم

الفرق بين struct و class

خلال تصفحي منتدى الفريق العربي للبرمجة لفتني البرنامج التالي؛ فقمتُ بتنفيذه:

```
#include <iostream>
#include <string>
#include <conio.h>

using namespace std;

struct C
{
protected:
        string str;
public:
        C()
        {cout << "Palestine ";}
        void SetName(string nstr)
        {str = nstr;}
        virtual string GetName(void)
        {return str;}
};

struct CCat : public C
{
public:
        void Meoo(void)
        {cout << "and: Al-Quds Open University.";}
        string GetName(void)
        {return ("\nand My city: "+str);}
        string operator+(C cre)
        {str = str+cre.GetName ();
        return str;}
};
void main()
{
        C cre;
        cre.SetName (" ");
        cout << cre.GetName () << "in my heart > ";
        CCat cat;
        cat.SetName ("Qalqilia");
        cout << cat.GetName () << endl;
        cat.Meoo ();
        cout << "\n\nSafa2 from ";
        string ope;
        ope = cat + cre;
        cout << ope << ": 0122010910128";
        getch ();
}
```

3 ناتج تنفيذ البرنامج أعلاه:



ودفعني الفضول لأقرأ أكثر حول الفرق بين التركيب والصنف
فوجدتُ أن خلاصة الأبحاث والتجارب تقول:

الفرق الوحيد بينهما هو أن:
مستوى الحماية الافتراضي لـــ struct هو public
ولـــ class المستوى الافتراضي private

ويمكن تغييره في الاثنين بكتابة الكلمات private و public بشكل صريح
بمعنى أنه يمكن تحويل الصنف لتركيب والتركيب لصنف.

والسبب لوجود كلمتين تؤديان نفس الوظيفة يعود **لأسباب تاريخية**،
فقد اعتمدت لغة C++ كلمة struct من باب **التوافقية** مع لغة سي التي لا تعرف الـــclass

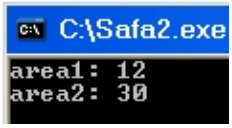
وأضافت C++ الـــ class لأن كلمة struct لم تعد صحيحة لغوياً وعلمياً
مع وجود مفهوم Abstract data type
الذي يضم الحقول والوظائف

وتعلق سلسلة SCHAUM'S صفحة 244 موضحة أن:
الفرق بين struct و class هو فرق **لا معنى له**؛ فكأنهما اسمان **لشخص واحد**
"So the difference between a class and a C++ struct is really jus cosmetic"
(السلسلة متوفرة على الإنترنت للتحميل بصيغة PDF)

`class C {`
`public:`
`int pub;`
`private:`
`int priv;`
`};`
	`struct S {`
	`public:`
	`int pub;`
	`private:`
	`int priv;`
	`};`

ثم قمتُ بتنفيذ عدة برامج وردت في الكتاب تطبيقاً على الصنف واستبدلت بكلمة class كلمة struct فكان ناتج التنفيذ في البرنامجين واحد! وأضع بين يديك هذا المثال:

```
#include <iostream.h>
#include <conio.h>


struct CRectangle
     {
     private:
     int width, height;
     public:
     CRectangle (int,int);
     int area () {return (width*height);}
     };


CRectangle::CRectangle (int a, int b)
             {
             width = a;
             height = b;
             }

void main ()
{

CRectangle A1 (3,4);
CRectangle A2 (5,6);

cout << "area1: " << A1.area();
cout << "\narea2: " << A2.area();

getch ();
}
```

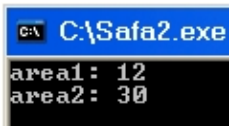


3 ناتج التنفيذ:

حتى مع عدم كتابة كلمة private يبقى ناتج التنفيذ هو ذاته

حيث أن البرنامج الأصلي هو:

```
#include <iostream.h>
#include <conio.h>

                اسم الصنف
class CRectangle
     {
     int width, height;  متغيرات منتمية - خاص
     public:
     CRectangle (int,int);  دوال منتمية - عام
     int area () {return (width*height);}
     };

  اسم الصنف    تقرير المجال      اسم الدالة              عواملها
CRectangle::CRectangle (int a, int b)
               {
               width = a;
               height = b;
               }

void main ()
{

CRectangle A1 (3,4);
CRectangle A2 (5,6);
                        اسم الكائن      اسم الدالة
cout << "area1: " << A1.area();
cout << "\narea2: " << A2.area();

getch ();
}
```



3 ناتج التنفيذ:

ليس هذا فحسب، بل مررت على المواقع المختصة بلغة C++

والتي تحوي أكواد خاصة بـــ classes

واستبدلت بكلمة class فيها كلمة struct فكان ناتج التنفيذ واحداً

فيمكن عمل بنّاء وهدّام خاص بالـــstruct

وأضع بين يديكم بعض الأمثلة:

- الــ struct تعمل عمل الــ class

```
#include <iostream.h>
#include <conio.h>

struct CRectangle {
  private:
    int x, y;
  public:
    void set_values (int,int);
    int area () {return (x*y);}
};

void CRectangle::set_values (int a, int b) {
  x = a;
  y = b;
}

void main () {
  CRectangle rect;
  rect.set_values (3,4);
  cout << "area: " << rect.area();

  getch ();
}
```

3 ناتج التنفيذ:

وكذلك الأمر لو كانت الكلمة class

- تعريف بنّاء constructor وهدّام destructor للـــ struct

```
#include <iostream.h>
#include <conio.h>

struct CRectangle {
  private:
    int *width, *height;
  public:
    CRectangle (int,int);
    ~CRectangle ();
    int area () {return (*width * *height);}
};

CRectangle::CRectangle (int a, int b) {
  width = new int;
  height = new int;
  *width = a;
  *height = b;
}

CRectangle::~CRectangle () {
  delete width;
  delete height;
}

void main () {
  CRectangle rect (3,4), rectb (5,6);
  cout << "rect area: " << rect.area() << endl;
  cout << "rectb area: " << rectb.area() << endl;

  getch ();
}
```

3 ناتج التنفيذ:

كل كود خاص بـــclass صادفني قمت بتنفيذه بكلمة struct والناتج واحد

أحببتُ أن أشارك بهذا البحث الصغير كتحضير لموضوع المحاضرة القادمة

وكنشاط خاص بمادة برمجة 1 / لغة C++

بارك الله بكم،

وجزاكم عنا كل خير.