المقدمة:

شاع استخدام المؤشرات (Pointers) في لغات البرمجة، بالخصوص C و C++، حيث لم يكن لها بديل لأداء الكثير من الوظائف و العمليات الحيوية التي لا غنى عنها في العديد من البرامج، مثل المصفوفات غير محددة الحجم (Dynamic Array) و مصفوفة المصفوفات أو ما يسمى بـ (Jagged Arrays) و منها مصفوفة السلاسل الحرفية (Strings Array) و كذلك في عملية (Late or Binding Dynamic) و غيرها من الحالات التي لا غنى عن استخدام المؤشرات (Pointers) فيها.

تعريف المؤشرات:

وهو عبارة عن عنوان للمتغير في الذاكرة ، إي كأنه يمثل رقم شقة في إسكان ، بصرف النظر عن محتوى الشقة وقيمة ما فيها.

مؤشرات الزيادة والنقصان في لغة C++:

من مزايا لغة السي أنها تستعمل الأداتين الحسابيتين ++ و -- لزيادة قيمة ١ أو إنقاصه والمثال التالي يبين طريقة الاستعمال:

a++؛ أو ++a ومعناه إضافة ١ إلي a ويمكن كتابته بصورة مكافئة على النحو a=a+1 .

وبصورة مشابهة يمكن إنقاص قيمة ١ من a على النحو:

a--؛ أو --a؛ وهو يكافئ الصورة:

a=a-1؛

ولكن هناك فرقاً في سرعة التنفيذ ، فالتعبير a++؛ أسرع من a=a+1؛ وهذه هي الفائدة من وراء استخدام مثل هذه الأدوات في لغة السي.

١- i) i=i+1 (++i) تعني

٢- i) i=i-1 (--i) تعني

٣- N) N=N+2 (N+ تعني

٤- X) X=X-100 (X-=100 تعني

المؤشرات pointers :

عند الإعلان عن المتغيرات يقوم الكمبيوتر بحجز مواقع لها في الذاكرة تسمى عناوين المؤشرات كسائر المتغيرات ولكنه يختلف عنها فيما يختزنه من البيانات فالمؤشر لا يختزن البيانات العادية مثل الأرقام أو الرموز ولكنه يختزن فقط عناوين الذاكرة ومن هنا جاء اسمه كمؤشر لأنه يشير مباشرة إلى أحد خانات الذاكرة فالمؤشر هو متغير يشير إلى عنوان متغير آخر في الذاكرة أو هو عبارة عن متغير يحتوي على عنوان متغير آخر في الذاكرة .

الإعلان عن المؤشرات :::

البوينتر كأي متغير في لغة السي ++ يجب الإعلان عنه لنتمكن من استعماله في

البرنامج
وكذلك قد يشير الى أي نوع من البيانات وتختلف طريقة الاعلان عن المؤشر بحسب نوع المتغير
الذي يشير إليه
كود: تحديد الكل

Int *npntr ؛
Double *ypntr ,*xpntr ؛

cpntr* Char؛

::: اسم المتغير يأتي مسبوقا دائما بالعلامة * التي تدل على أنه مؤشر :::
استعمال المؤشرات :::
بعد الاعلان عن المؤشر في بداية البرنامج تأتي الخطوة التالية وهي تخصيصه ليشير الى عنوان متغير معين ، وهذا يتم عن طريق ذكر اسم المؤشر فقط وتخصيص اسم متغير له شريطة ان يكون اسم المتغير مسبوقا بالعلامة &
y& = Ypntr
العلامة & يجب كتابتها قبل اسم المتغير لتدلنا على عنوانه في الذاكرة

كيف و متى تستخدم المؤشرات (Pointers)؟

استخدام المؤشرات (Pointers) في C# غير مستحسن، إلا في حالة أن يكون استخدامها يزيد من سرعة أداء برنامجك بنسب معقولة، أو أن تكون بحاجة لاستخدام مكتبات ربط ديناميكي (DLL) و ما شابهها أو كائنات (COM) و غيرها، و التي لا تكون خاضعة لنظام إدارة الذاكرة التابع لـ .Net ففي هذه الحالات يكون استخدام المؤشرات (Pointers) منطقيا أو أمرا لا بد منه، أما ما سوى ذلك فلا، لأن C# و إضافة إلى أنها وفرت على المبرمج أداء العمليات الاعتيادية المرتبطة بالذاكرة، و التي كان عليه إنجازها يدويا في C و C++، و بسرعة تضاهي سرعة برامج هاتين اللغتين، فهي لا تعاني من البطء الذي تعاني منه برامج لغات أخرى مثل جافا (Java)،.

عند استخدام المؤشرات (Pointers) في C#، يجب استخدامها ضمن العبارة (unsafe) و هي كلمة محجوزة (Reserved word or Keyword) تحدد جزء الشيفرة (Code) الذي يتضمن استخداما للمؤشرات (Pointers)، و هذه الكلمة يمكن استخدامها بمفردها أو عند تعريف الأعضاء (Members) سواء كانت من الخصائص (Properties) أو الوظائف (Functions) أو المشيدات (Constructors) أو غيرها و كذلك عند تعريف الفئات (Classes) و ما إلى ذلك،

المؤشر هو طريقة للاعلان عن المتغيرات من أى نوع حيث تمكن المبرمج من التعامل مع عنوان المكان المحجوز فى الذاكرة مباشرة وليس القيمة.

ويتم الاعلان عن مؤشر الى أى متغير بوضع العلامة * قبل اسم المتغير كما فى الصور التالية:

int *a

float *b

char *c

وفى هذه الصور يتم الاعلان عن متغيرات من بعض أنواع البيانات، ولكن يسبق المتغير العلامة * مما يجعل المتغير يشير الى عنوان المكان المحجوز فى الذاكرة وليس القيمة،

مزايا استعمال المؤشرات

١- زيادة سرعة تنفيذ خطوات البرنامج حيث نتعامل مع عنوان المكان المحجوز مباشرة

٢- اعادة أكثر من قيمة من الدوالي Return more than one value

٣- سهولة التعامل مع أنواع البيانات المختلفة مثل المصفوفات

السجلات Structures

السجل عبارة عن مجموعة من البيانات مختلفة النوع تحت اسم واحد ،ويستخدم فى تمثيل بيانات شيئ ما تمثيلا كاملا ،مثل سجل بيانات طالب ،سجل بيانات صنف وهكذا.

أنواع المؤشرات (Pointers):

1. مؤشرات ثابتة (كما في المتغيرات التي تحمل قيمة ثابت.
2. مؤشرات متغيرة: و هذه المؤشرات هي البداية الى عالم البرمجة الديناميكية.

ثانيا: تعريف مؤشر.

يمكنك تعريف المؤشر باستخدام الصيغة التالية:

typeName * pointerName;

و يمكن وضعه مبدئيا ليشير الى متغير ما مثلا كقيمة ابتدائية:

typeName * pointerName = &variableFromSameType;

حيث علامة & تعطي العنوان في الذاكرة للمتغير variableFromSameType

أما علامة * فتستخدم مع المؤشر لكي نستطيع التعامل مع محتوى الذاكرة هناك.

للمزيد من التوضيح الرجاء قراءة الملف الملحق..

المؤشرات في C# على نوعين:

- void *: و هو المؤشر (Pointer) غير محدد النوع.
- type *

و (type) هو أحد الأنواع التالية:

- أنـواع القـيم (Value Types) و هـي: bool, sbyte, byte, short، ushort, int, uint, long, ulong, char, float, double, decimal, enum
- أنـواع المؤشـرات (Pointer Types): أي المؤشـرات علـى المؤشـرات (Pointer Pointer to)
- الأنـواع المعرفـة مـن قبـل المسـتخدم (User-Define Types): و هـي أي أنـواع مـن قبيـل التراكيـب أو السـجلات (Structures) و ليسـت الفئـات (Classes)، فالأولى تعتبر من أنواع القيم (Value Types) و التـي يمكن الإشارة إليها (Pointing to)، و الثانية من الأنواع المرجعية (Reference Types) و التـي لا يمكن الإشـارة إليها بـأي حـال للسبب المذكور فـي الفقرة السـابقة، و لكـن تسـتثنى المصـفوفات (Arrays) مـن الأنـواع المرجعيـة (Reference Types) التـي لا يمكـن الإشـارة إليهـا، حيـث إن الإشـارة للمصفوفات (Arrays) ممكنة لكن بشرط سنأتي على ذكره لاحقا. و أخيرا يجب الانتباه إلى أن التراكيب أو السجلات (Structures) يجب أن لا تحتوي في تركيبها على أي من الأنواع المرجعية (Reference Types) و إلا لن يكون بالإمكان الإشارة إليها.

- (*) هي جزء من اسم النوع (Type Name) و ليست سابقة لاسم المتغير (Name Variable) كمـا هـو الحـال فـي C و C++، و يتضـح ذلـك فـي المثـال: (pC1, pC2 * char) حيث تـم تعريـف مؤشـرين مـن نـوع (char *)، أما في C و C++ فنتيجة العبارة السابقة هي تعريف مؤشر (Pointer) لـ (char) و هو (pC1) و متغير من نوع (char) هو (pC2)
- تستخدم (*) أو ما يعرف بـ (Pointer Indirection Operator) للحصول على القيمـة التـي يشـير إليهـا المؤشـر (Pointer) مثلمـا هـو الحـال فـي C و C++، لكـن الاخـتلاف فـي C# يكمـن فـي أنـه لا يمكـن الحصـول علـى قيمـة المؤشر نفسه - عنوان المتغير الذي يشير إليه -
- المؤشر من نوع (T *) يحتوي - كقيمة له - عنوان متغير من نوع (T)
- إن كان نوع الكائن (Object) من أنواع المؤشرات (Pointer Types) يتم استخدام المعامل (Operator) (->) بدل المعامل (Operator) (.) للوصول إلـى أعضـاء هـذا الكـائن كمـا هـو واضـح فـي المثـال: (pMyByte->ToString())
- لأن المصـفوفات (Arrays) مـن الأنـواع المرجعيـة (Reference Types) فهي تخضع نظام الإدارة الآلية للذاكرة، و بالتحديد لـ (GC)، و هـي معرضـة فـي أي وقت لتغيير عنوانها - عنوان العنصر الأول فيها - أو للإزالـة نهائيـا مـن الـذاكرة - عنـد انتفـاء الحاجـة إليهـا - و لـذلك عنـد الإشـارة للمصـفوفات (Pointing to Arrays) نسـتخدم العبـارة (fixed) التـي تحمـي الكـائن (Object) - مؤقتـا - مـن عمليـات (GC)، و تثبتـه فـي محلـه. و لأن اسـتخدام العبارة (fixed) لا يتناسب مع طبيعة و هدف (GC)، لا يمكن حجز و تثبيت

الكائن (Object) إلا لفترة قصيرة، و لإنجاز عمليات سريعة تتطلب وجود الكائن (Object) في محله حتى الانتهاء منها. و هنا يجب الانتباه إلى أن المؤشر (Pointer) الذي يتم تعريفه في العبارة (fixed) ثابت القيمة - المكان الذي يشير إليه -، و لا يمكن تغيير قيمته.

الخلاصة:-

و خلاصة القول أن المؤشرات (Pointers) في C# لا تختلف كثيرا عن نظيراتها في C و C++، و لكن و على الرغم من توفر ميزة استخدام المؤشرات (Pointers) في C#، إلا أن تجنب استخدامها أولى من استخدامها، ما لم تكن تؤدي للمبرمج أعمالا و تنجز له أمورا بحيث لا يمكن الاستغناء عنها بغيرها.

- مع تحيات المهندس/نبيل مصلي

.Email:-nabil299@gmail.com