المفاهيم البرمجيّة الشائعة في لغة C:

لنلقِ نظرة على كيفيّة وضع بعض المفاهيم البرمجيّة الشائعة في حيّز التنفيذ في الكود البرمجي الخاص بك بلغة C. التالي هو ملخّص سريع لهذه المفاهيم:

الاقترانات (الدوال) Functions: كما ذُكر سابقاً، إنّ الاقتران هو جزء من كود برمجي يمثّل شيئاً ما يجب على الحاسوب فعله عند عمل البرنامج. بعض اللغات تدعو هذا بالاقترانات التركيبية **structures methods**، على الرغم من أنّ مبرمجي لغة الـC يستخدمون المصطلح ذاته بشكل اعتيادي. يمكن لبرنامجك أن يعرّف عدّة اقترانات ويستدعيها من خلال اقترانات أخرى. سنأخذ نظرة أقرب على بنية الاقترانات في لغة البرمجة C لاحقاً.

المتغيرات Variables: قد تحتاج للمرونة في بعض الأحيان لتشغيل البرنامج دون الحاجة لمعرفة ما هي هذه القيم مع الزمن. وكما في لغات البرمجة الأخرى، تسمح لك لغة البرمجة C باستخدام المتغيرات عندما تحتاج إلى تلك المرونة. على غرار المتغيرات في الجبر، يكون المتغير في البرمجة الحاسوبيّة هو عبارة عن نائب يستخدم من أجل تخزين قيمة ما لا تعلمها أو لم يتم إيجادها بعد.

نوع البيانات Data types: بهدف حفظ البيانات في الذاكرة في أثناء عمل البرنامج، ولمعرفة ما هي العمليات التي بإمكانك تنفيذها على المتغيرات المستخدمة لتخزينها، إنّ لغةً برمجيةً مثل C تعرّف أنواعَ بيانات معيّنة لتدركها. وكل نوع بيانات في لغة C له حجم ذاكرة معيّن، مُقاس بالقيمة الثنائية (البيت- bit) أو (بالبايت- byte) وهو مجموعة عددية من القيم الثنائية، بالإضافة إلى مجموعة معيّنة من القواعد عمّا تمثّله قيمها. لاحقاً، سنرى مدى أهمية اختيار نوع البيانات الصحيح لمهمّة ما عندما تستخدم لغة C.

العمليّات Operations: في لغة C، بإمكانك تنفيذ عمليات حسابيّة (كالجمع) على الأعداد وسلسلة من العمليّات (كالسلسلة الرياضيّة) أو سلاسل من المحارف. كما إن للغة C عمليات مدمجة بها مصمّمة خصيصاً لأغراض قد تريد تنفيذها على البيانات الخاصّة بك. حيث عندما نفحص أنواع البيانات في لغة C، سنأخذ نظرة موجزة على العمليّات أيضاً.

التكرارات Loops: أحد أكثر الأشياء الأساسيّة التي يرغب المبرمج في تنفيذها هو إعادة تنفيذ فعلٍ ما لعددٍ معيّنٍ من المرّات بناءً على شروط محدّدة تأتي في أثناء عمل البرنامج. هناك جزء من الكود البرمجي مصمّم ليتكرر بناءً على شروط مُعطاة ويدعى "التكرار **Loop**"، ولغة البرمجة C تؤمّن جمل التكرارات الشائعة وهي:

while, do/while, for, continue/break and goto

تتضمّن لغة C أيضاً التعابير الشرطيّة الشائعة:

If/ else

Switch/ case

تراكيب البيانات Data structures: عندما يكون على البرنامج الخاص بك معالجة العديد من البيانات، فيم أنت بحاجة إلى فرز أو البحث ضمن تلك البيانات، فمن المحتمل أنّك ستستخدم نوعًا يسمى بتراكيب البيانات. تراكيب البيانات هي طريقة هيكليّة لتمثيل عدّة اجزاء من البيانات من نفس النوع. وتراكيب البيانات الأكثر شيوعاً هي المصفوفة، والتي هي عبارة عن قائمة مفهرسة من حجم مُعطى. تمتلك لغة C مكتبات متوفّرة لمعالجة بعض أنواع البيانات الشائعة، مع ذلك يمكنك دائماً كتابة الاقترانات وضبط التراكيب الخاصّة بك أيضاً.

عمليّات ما قبل المعالجة: في بعض الأحيان، قد ترغب بإعطاء مترجم الكود البرمجي **compiler** بعض التعليمات حول أمورٍ لتنفيذها مع الكود الخاص بك قبل ترجمته ليصبح جاهزًا للتنفيذ على الحاسوب. هذه العمليّات تتضمّن استبدال قيمة ثابت معيّن، وتتضمّن كودًا من مكتبات لغة البرمجة C (والتي رأيتها في نموذج الكود البرمجي سابقاً).

كما إنّ لغة C تتطلّب من المبرمجين معالجة بعض المفاهيم والتي بسطتها العديد من لغات البرمجة أو جعلتها مؤتمتة حيث تتضمّن المؤشرات، إدارة الذاكرة، وجمع النفايات (حذف البيانات التي لم يعد لها استخدام من قِبل الذاكرة). ما يلي، سيغطي الأمور المهم معرفتها حول هذه المصطلحات أثناء البرمجة باستخدام لغة البرمجة C.

قد تبدو هذه النظرة العامّة السريعة على المفاهيم مربكة إن لم تكن مبرمجاً بشكل مسبق. فقبل أن تنتقل إلى استعراض الدليل البرمجي الكثيف بلغة C، دعنا نلقي نظرة سريعة على المفاهيم الأساسيّة ضمن هذه القائمة في الأعلى، ولنبدأ بالاقترانات.

الاقترانات في لغة البرمجة C:

إنّ أغلب لغات البرمجة الحاسوبيّة تسمح لك بإنشاء اقترانات من بعض الأنواع، حيث تسمح لك الاقترانات باختصار برنامج طويل إلى أقسام لها أسماء، وبالتالي سيتسنى لك إعادة استخدام هذه الأقسام ضمن البرنامج من خلال استدعائها باسمها المخصص لها.

إنّ مبرمجي بعض اللغات، وبالأخص أولئك الذين يستخدمون تقنيّات البرمجة التي تسمح بتشكيل البرنامج كمجموعة من الأغراض التي يمكن أن يتم التعامل معها بأسلوب الوحدات **object-oriented programming**، يستخدمون المصطلح: طريقة **method**، عوضاً عن اقتران **function**.

الاقترانات يمكنها أن تستقبل العوامل المرسلة **parameters** وإعادة النتيجة. إنّ القسم من الكود البرمجي الذي يشمل الاقتران هو تعريف الاقتران **function definition**.
التالي هو بنية أساسيّة لتعريف اقتران:

```
<return type> <function name>(<parameters>)
{
<statements>
return <value appropriate for the return type>;
}
```

شكل 1

كحد أدنى، برنامج بلغة C يملك اقترانًا وحيدًا يدعى الاقتران الرئيسي **main**. سيبحث المترجم **compiler** عن البرنامج الرئيسي كنقطة البداية للبرنامج، حتّى ولو استدعى الاقتران الرئيسي اقترانات أخرى ضمنه. الكود البرمجي التالي هو الاقتران الرئيسي الذي رأيناه في البرنامج بلغة C الذي تفحّصناه سابقاً. الاقتران الرئيسي يعيد قيمة صحيحة **integer**، ولا يأخذ عوامل، ويملك تعليمتين، إحداها هي تعليمة إعادة القيمة **return** :

```
int main()
{
printf("This is output from my first program!\n");
return 0;
}
```

شكل 2

الاقترانات الأخرى غير الاقتران الرئيسي تملك تعريفًا واستدعاءً لمرّة واحدة أو أكثر. استدعاء الاقتران هو عبارة أو جزء من عبارة ضمن اقتران آخر. يستدعي الاقتران اسم الاقتران الذي يود استدعاءه متبوعاً بقوسين. فإذا كان الاقتران ذا عوامل، فيتوجّب على عبارة الاستدعاء أن تحتوي على قيم موافقة لتناسب هذه العوامل. هذا الجزء الإضافي من عبارة استدعاء الاقتران يدعى تمرير العوامل للاقتران.

ولكن ما هي العوامل؟ عامل ما لاقتران معيّن هو متغير من البيانات من نوع محدّد، والتي يتطلّبها الاقتران ليقوم بعمله. الاقترانات في لغة البرمجة C يمكن أن تقبل عددًا لا محدودًا من العوامل، تدعى في بعض الأحيان بـ **arguments**. كل عامل يُضاف إلى اقتران ما يجب أن يعيّن شيئين اثنين: نوع بياناته واسم متحوّله (المتغيّر) ضمن قطعة الاقتران. عند تعريف عدّة عوامل يتم الفصل بينها بفاصلة. في الاقتران التالي، هناك عاملان، وكلاهما من النوع **integer** (عدد صحيح):

```
int doubleAndAdd(int a, int b)
{
return ((2*a)+(2*b));
}
```

شكل 3

والآن، لنتابع تفحّصنا للاقترانات عبر النظر في كيفيّة تناسبها ضمن برنامج أكبر بلغة C.

وضع نماذج بدائيّة للتوابع Function Prototypes:

في لغة C، بإمكانك وضع تعريف الاقتران في أي مكان ضمن البرنامج (عدا ضمن اقتران آخر). الشرط الوحيد الذي يتوجب عليك فعله هو إعلامُ المترجم **compiler** مُقدماً أنّ الاقتران موجود في مكان ما لاحقًا في الكود البرمجي، ستقوم بذلك مع وضع نموذج الاقتران عند بداية البرنامج. نموذج الاقتران هو عبارة تشبه تلك الموجودة في السطر الأول من تعريف الاقتران.

في لغة C، لست مضطراً لإعطاء أسماء للعوامل في نموذج الاقتران، فقط تذكر نوع البيانات. العبارة التالية هي ما سيبدو عليه نموذج الاقتران ذو الاسم **doubleAndAdd**:
int doubleAndAdd(int, int);

تخيّل نماذج للاقترانات كقائمة حزم لبرنامجك، سيقوم المترجم بفك الحزم وتجميع البرنامج الخاص بك تماماً كما تقوم أنت بفك حزم وترتيب رف كتب جديد. تساعدك قائمة الحزم على التأكّد بأنك تملك كل القطع التي تحتاجها في الصندوق قبل أن تبدأ بترتيب رف الكتب. يستخدم المترجم نماذج الاقترانات بنفس الطريقة قبل أن يبدأ بتجميع برنامجك.

إذا كنت قد أعطيت اهتماماً لنموذج البرنامج بلغة C الذي تفحّصناه سابقاً من حيث التجربة، فعليك بفتح الملف والتعديل عليه لإضافة نموذج اقتران. ابدأ أولاً بتعريف اقتران جديد، واستدعاء الاقتران **doubleAndAdd** الموضّح هنا. من ثم، ترجم وشغّل برنامجك كما في السابق لترى كيف يعمل الكود الجديد. كما يمكنك استخدام الكود البرمجي التالي كدليل لتجرّب ذلك:

```
#include <stdio.h>
int doubleAndAdd(int, int);
int main()
{
printf("This is output from my first program!\n");
printf("If you double then add 2 and 3, the result
is: %d \n", doubleAndAdd(2,3));
return 0;
}
int doubleAndAdd(int a, int b)
{
return ((2*a)+(2*b));
}
```

شكل 4

إلى هنا، كنا قد ألقينا نظرة على بعض العناصر البنيويّة الرئيسيّة في برنامج ما بلغة C. دعنا الآن نلقي نظرة على أنواع البيانات التي بإمكانك العمل معها في برنامج بلغة C والعمليّات التي يمكنك تنفيذها على تلك البيانات.

تصاريح الاقترانات Function declarations:

في لغة الـ C، على الأغلب ستسمع المصطلح تصريح الاقترانات أكثر من نموذج بدائي للاقتران **function prototype**، وخاصّة بين مبرمجي لغة C الأكبر عمراً، مع ذلك فإنّنا نستخدم المصطلح **function prototype** في هذه المقالة، لأنّها تملك اختلافًا مهمًّا. أوّلاً، إنّ تصريح الاقتران لا يتطلّب أيّة عوامل، كذلك نوع القيمة المُعادة، حيث أنّ اسم الاقتران وزوجًا من الأقواس الفارغة فقط سيكونان كافيين. نموذج الاقتران يعطي المترجم معلومات إضافيّة هامّة للغاية عبر تضمين عدد ونوع البيانات الخاصّة بالعوامل التي سيستدعيها. أصبحت النماذج المنهج العملي الأفضل بين كاتبي الكودات في يومنا هذا سواء في لغة C أم في لغات البرمجة الأخرى.

نوع البيانات والعمليّات في لغة البرمجة C:

من وجهة نظر حاسوبك، البيانات هي عبارة فقط عن سلاسل من الواحدات والأصفار يمثلون حالات عمل **on** وإيقاف **off** للبتات الثنائيّة الالكترونيّة على الأقراص الصلبة أو ضمن معالج الحاسوب أو الذاكرة. وحده الكيان البرمجي الذي تشغّله على الحاسوب يجعل من هذه المليارات من الوحدات الثنائيّة ذات معنى.

لغة C هي واحدة من بضعة لغات برمجيّة عالية المستوى التي بإمكانها التعامل بسهولة مع البيانات على مستوى البت بالإضافة إلى تفسير البيانات بناءً على نوع بيانات مُعطى.
نوع البيانات هو مجموعة صغيرة من القواعد التي تشير إلى كيفيّة جعل سلسلة من البتات ذات مغزى. نوع البيانات له حجم محدّد بالإضافة إلى طريقته الخاصّة في تنفيذ العمليّات (كالجمع والضرب) على البيانات من ذلك النوع.

في لغة الـ C، حجم نوع البيانات مرتبط بالمعالج الذي تستخدمه. على سبيل المثال، في **C99**، جزء بيانات من النوع **integer** ـ اختصاراً (**int**) هي بطول 16 بت ضمن معالج 16 بت، بينما من أجل المعالجات 32 بت و64 بت تكون بطول 32 بت.

شيء آخر مهم لمبرمجي الـ C، هو معرفة كيف تقوم اللغة بمعالجة البيانات المؤشّرة وغير المؤشّرة. نوع المؤشّر يعني أنّ واحدًا من بتاتها محجوز كمؤشّر ليشير فيما إذا كان رقمًا موجبًا أو سالبًا. لذا، في حين أنّ النوع int غير المؤشّر في نظام 16 بت يمكنه حمل الأرقام بين 0 و65,535، أمّا المؤشّر في نفس النظام يمكنه حمل الأرقام بين ـ32,768 إلى 32,767. فإذا أنتجت عمليّة ما متغيرًا من النوع الصحيح خارج مجاله، فعلى المبرمج معالجة حالة البينات الفائضة **overflow** ضمن كود برمجي إضافي.

ونظراً لهذه القيود والخصوصيّات الخاصّة بالنظام في نوع البيانات والعمليّات في لغة C، فعلى مبرمجي لغة C اختيار نوع بياناتهم في لغة الـ C، بمعنى تلك التي بنيت بلغة البرمجة C. انظر إلى دليلك المفضّل في البرمجة بلغة C لتحصل على قائمة كاملة من نوع البيانات في لغة C ومعلومات مهمّة حول كيفيّة تحويل البيانات من نوع لآخر.

كما إنّه بإمكان مبرمجي الـ C إنشاء تراكيب البيانات، والتي تدمج نوع بيانات بدائيّة ومجموعة من التوابع التي تعرّف كيف يمكن للبيانات أن تكون منظّمة ومعالجة. على الرغم من أنّ استخدام تراكيب البيانات يعتبر موضوع برمجة مُتقدّمة وخارج نطاق هذا المقال، إلّا أنّنا

سنلقي نظرة على التراكيب الرئيسيّة الأكثر شيوعاً، المصفوفات، حيث إنّ المصفوفة هي عبارة عن قائمة افتراضيّة تتضمّن قطع البيانات التي هي جميعها من نفس النوع . وحجم المصفوفة لا يمكن أن يتغيّر، مع إنّه يمكن نسخ محتوياتها إلى مصفوفة أخرى أكبر أو أصغر.

على الرغم من أنّ المبرمجين غالباً مايستخدمون مصفوفات الأرقام، إلّا أنّ مصفوفات الحروف، والمسماة بالسلاسل، تملك الخواص الأكثر تفردًا. حيث إنّ السلسلة تسمح لك بحفظ شيء يمكنك التلفظ به (مثلاً "**Hello**") ضمن سلسلة من الحروف، والتي يمكن أن يقرأها برنامجك بلغة الـ C من المستخدم أو يطبعها على الشاشة. إنّ معالجة السلاسل لها عمليّات فريدة مشابهة، حيث إنّها تمتلك مكتبة C مخصّصة لها (**string.h**) مع توابع السلاسل العاديّة.

إنّ العمليّات المدمجة في لغة C هي العمليّات النمطيّة التي قد تراها في غالب لغات البرمجة. وعندما تدمج عدّة عمليّات ضمن عبارة واحدة، عليك التأكّد من معرفتك بأسبقيّة المُشغّل، أو الترتيب الذي سينفّذ عبره البرنامج كل عمليّة ضمن التعبير الرياضي. على سبيل المثال، (2+5*3) تساوي 17، لأنّ C ستنفّذ عمليّة الضرب قبل الجمع إلّا إذا كانت هنالك أقواس تشير إلى غير ذلك.

إذا كنت تتعلّم لغة البرمجة C، اجعل التآلف مع جميع أنواع بياناتها الأوليّة والعمليّات والأسبقيّة للعمليّات في نفس العبارة من أولوياتك. أيضاً، قم بتجربة عدّة عمليّات على متحولّات وأرقام من أنواع بيانات مختلفة.

عند هذه المرحلة، تكون قد خدشت سطح بعض أساسيّات لغة البرمجة C. لاحقًا، سنلقي نظرة حول كيف تسمح لك لغة C بكتابة برامج دون الحاجة للبدء من الأساسيّات كل مرّة.

- التاريخ: 2016-08-09
- التصنيف: تكنولوجيا

#البرمجة #لغات البرمجة #البرمجة بلغة c

المصطلحات

- **الأيونات أو الشوارد (Ions)**: الأيون أو الشاردة هو عبارة عن ذرة تم تجريدها من الكترون أو أكثر، مما يُعطيها شحنة موجبة.وتسمى أيوناً موجباً، وقد تكون ذرة اكتسبت الكتروناً أو أكثر فتصبح ذات شحنة سالبة وتسمى أيوناً سالباً

المصادر

- how stuff works

المساهمون

- ترجمة
  - محمد اسماعيل باشا
- مُراجعة
  - أمجد هواش
- تحرير
  - أنس الهود
  - بنان محمود جوابره
- تصميم
  - علي كاظم
- نشر
  - سارة الراوي