الفصل 3 : التعليمات الشرطيّة

1. مقدمة

البرنامج هو مجموعة من التعليمات. حيث يتمّ تنفيذ معظم هاته التعليمات على الترتيب الذي ظهرت عليه. فبعد الانتهاء من تنفيذ تعليمة ما، يتمّ الانتقال إلى التعليمة التي تليها مباشرة (séquentiel). وغالبًا ما تكون التعليمات مفصولة بفاصلة منقوطة « ; ». لكن، هناك بعض التعليمات التي من شأنها تغيير تدفق البرنامج، تُدعى بعمليات التحكم (structures de contrôles). مثل: البنية الشرطيّة، الحلقات، استدعاء الدوال، وعمليات الوصل الغير شرطيّة. وتأتي البنية الشرطيّة في ثلاث أنواع هي : البنية الشرطيّة البسيطة (si alors)، البنية الشرطيّة المركّبة (si alors sinon)، وبنية الاختيارات المتعدّدة (selon).

2. التعليمة الشرطيّة البسيطة si alors

في البرمجة، غالبا ما نصادف حالات يجب فيها تحديد ما إذا كان يجب تنفيذ بعض التعليمات أم لا، بناءً على ما إذا كان الشرط صحيحًا أم لا. فمثلا، في وصفة تحضير الحلوى، نجد تعليمات من الشكل: إذا لديك لوز فأضفه للوصفة، أو إذا كنت تحب الليمون فاضف قليلا. حيث سيتغيّر تدفّق تنفيذ البرنامج بتغيّر المدخلات. وللتعبير عن الشرط في البرمجة، نستعمل الاختبار si…alors (إذا كان…اذن…)، وهو ابسط التعليمات الشرطيّة. ويتكون من جزئين هما:

- الشرط: وهو عبارة عن تعبير expression من نوع منطقي، تكون قيمته إمّا صحيح vrai، أو خاطئ faux.
- كتلة تعليمات: يتمّ تنفيذها في حالة الشرط صحيح vrai، أو تُجاوزها في حالة الشرط خاطئ faux.

2.1. الصيغة

الخوارزم	لغة C
Si شرط **Alors** كتلة التعليمات **FinSi** بقية التعليمات	**if** (شرط) **{** كتلة التعليمات **}** بقية التعليمات

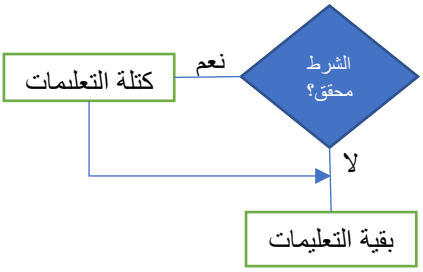
حيث أنّ الكلمات **si**، **alors** و**finSi** أو **fsi** هي كلمات محجوزة في الخوارزم. كذلك الحال بالنسبة لـ if في C. ويكون الشرط دوما في الخوارزم بين كلمتي si وalors، أمّا في C، فيكون دوما بين قوسين (). ولبناء الشرط نستعمل عمليّات المقارنة (<، >، =، ≠، ...) والعمليّات المنطقية (et، ou، non، ...).

تكون التعليمات التابعة لـ if في C محصورة بين حاضنتين {}، ويمكن حذفهما إذا كانت لا تحتوي إلّا على تعليمة واحدة ({} اختيارية). وإذا وجدنا مجموعة من التعليمات بعد if، ولم نجد الحاضنتين، فهذا يعني أنّ التعليمة الأولى فقط هي التي تتعلق بالشرط، أمّا باقي التعليمات فستنفذ دوما ولا علاقة لها بالشرط. ولكن، إذا كان هناك أكثر من تعليمة داخل الحاضنتين {} فإنّها، في هذه الحالة، تكون إلزاميّة.

ملاحظة

- في C يُعبر عن النوع المنطقي بعدد صحيح int. حيث يُعبّر عن خاطئ faux بـ 0، وعن صحيح vrai بأيّ عدد يختلف عن 0.
- لا توجد ";" بعد}.

2.2. الهيكل التنظيمي algorigramme



2.3. التنفيذ

وتتمّ عمليّة تنفيذ التعليمة الشرطيّة بحساب عبارة الشرط، والتي تكون نتيجتها من نوع منطقي Boolean، فإذا كانت النتيجة صحيحة vrai، يتمّ تنفيذ كتلة التعليمات التي بين كلمة alors وfin si في الخوارزم، أو بين {} في C، ثم يتمّ تنفيذ بقية تعليمات البرنامج. أمّا إذا كانت النتيجة خاطئة faux، فيتمّ تخطّي التعليمات التي بين كلمة alors وfinSi، ويتمّ تنفيذ بقية تعليمات البرنامج مباشرة.

2.4. مثال

اكتب البرنامج الذي يقرا عددا صحيحا، ثم يظهر تحذيرا إذا كان سالبا، ثم يظهر لنا مربّعه.

الخوارزم	لغة C	الشاشة

algorithme racine
var x :entier
début
écrire("ادخل عددا صحيحا")
lire(x)
Si x<0 **Alors**
écrire("العدد سالب")
FinSi
écrire("مربعه هو" , x*x)
fin
	#include <stdio.h>
	int main()
	{
	int x ;
	printf("ادخل عددا صحيحا\n") ;
	scanf("%d", &x) ;
	if (x<0)
	{//يمكن الاستغناء عنها
	printf("العدد سالب\n") ;
	}
	printf("مربعه هو%d" , x*x) ;
	}
		C:\Users\Moi\Documents\
		entrer un nbr
		2
		son carre est 4
		**********
		entrer un nbr
		-3
		le nbr est negatif
		son carre est 9

3. التعليمة الشرطية المركّبة si alors sinon

في التعليمة الشرطية البسيطة تحدّد si ما الذي يجب فعله إذا كان الشرط محققا، ولكنّها لا تحدّد ما يجب فعله إذا كان خاطئا. لكن، في بعض الأحيان، يجب تحديد ما الذي يجب فعله في كلتا الحالتين. لذلك، تأتي التعليمة sinon si (إذا كان.. إذن.. وإلّا...) والتي هي امتداد لـ si البسيطة. وتتكون التعليمة الشرطية المركّبة sinon si من ثلاث أجزاء هي:

- الشرط: وهو عبارة عن تعبير expression من نوع منطقيّ، تكون قيمته إمّا صحيح vrai، أو خاطئ faux.
- كتلة تعليمات الأولى: ويتمّ تنفيذها في حالة الشرط صحيح vrai، أو تجاوزها في حالة الشرط خاطئ faux.
- كتلة تعليمات الثانية: ويتم تنفيذها في حالة الشرط خاطئ faux، أو تجاوزها في حالة الشرط صحيح vrai.

3.1. الصيغة

الخوارزم	لغة C

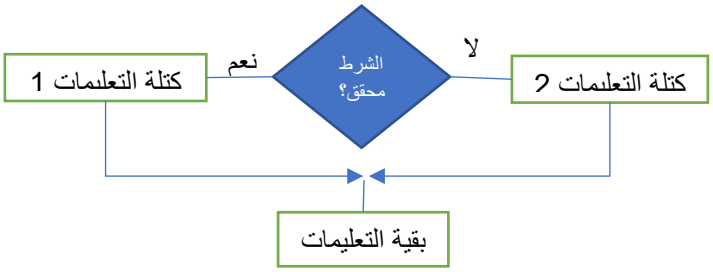
Si شرط **Alors**
كتلة التعليمات 1
sinon
كتلة التعليمات 2
FinSi
بقية التعليمات
	if (شرط)
	{
	كتلة التعليمات 1
	}
	else
	{
	كتلة التعليمات 2
	}

	بقية التعليمات

حيث أنّ الكلمة **sinon** هي كلمة محجوزة في الخوارزم. كذلك الحال بالنسبة لـ **else** في C.

يمكن الاستغناء عن {} في C إذا كانت لا تحوي إلّا تعليمة واحدة. وإذا وجدنا مجموعة من التعليمات بعد if أو بعد else، ولم نجد الحاضنتين، فهذا يعني أنّ التعليمة الأولى فقط هي التي تتعلق بـ if أو بـ else.

3.2. الهيكل التنظيمي algorigramme



3.3. التنفيذ

تتمّ عمليّة تنفيذ التعليمة الشرطيّة بحساب عبارة الشرط، والتي تكون نتيجتها من نوع منطقي Boolean، فإذا كانت النتيجة صحيحة vrai، يتمّ تنفيذ كتلة التعليمات الأولى التي بين كلمة alors وsinon في الخوارزم، أو بين {} التي قبل else في C، ثمّ يتمّ تنفيذ بقيّة تعليمات البرنامج. أما إذا كانت النتيجة خاطئة faux، فيتمّ تنفيذ كتلة التعليمات الثانية التي بين كلمة sinon وfinSi في الخوارزم، أو بين {} التي بعد else في C، ثمّ يتمّ تنفيذ بقيّة تعليمات البرنامج.

3.4. مثال

اكتب البرنامج الذي يحسب القيمة المطلقة لعدد صحيح، ثمّ يظهرها على الشاشة.

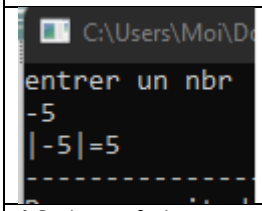
الخوارزم

```
algorithme absolue
var x, y :entier
début
  écrire("ادخل عددا صحيحا")
  lire(x)
  Si x>=0 Alors
    y←x
  sinon
    y←-x
  FinSi
  écrire("|" , x ,"|=",y)
fin
```

لغة C

```
#include <stdio.h>
int main(){
int x, y ;
printf("ادخل عددا صحيحا\n") ;
scanf("%d", &x) ;
if ( x>=0 )
{//يمكن الاستغناء عنها
   y=x ;
}
else
{//يمكن الاستغناء عنها
   y=-x ;
}
printf(("|%d|=%d", x, y)) ;
}
```

الشاشة




```
if ( x>=0 )
   y=x ;
else
   y=-x ;
```

3.5. الإسناد الشرطيّ في C

إذا كان لدينا متغيّر v يأخذ أحد القيمتين v1 او v2 حسب الشرط b، أي:

```
if (b)
v=v1 ;
else
v=v2 ;
```

في هذه الحالة، يمكن استعمال العمليّة : ? وصيغتها كالتالي:

```
condition ? expression_vrai : expression_faut
```

condition شرط من نوع منطقي

expression_vrai العبارة التي ترجعها في حالة الشرط صحيح.

expression_faut العبارة التي ترجعها في حالة الشرط خاطئ.

مثال

```
v=b ? v1 :v2 ;
result= moy>=10 ? "Admis" : "Ajourné" ;
```

3.6. امتداد sinon si

يمكن استعمال sinon si لاختبار عدّة حالات، واختيار المعالجة المناسبة لكلّ حالة. فمثلا: لمعرفة إذا كان الطالب ناجحا أم لا، هناك عدّة حالات. إمّا أن يكون ناجحا بدون تعويض، أو ناجحا بالتعويض، أو ناجحا بديون، أو راسبا. ولمعرفة ذلك، يجب معاينة معدّليْ السداسيين الأول والثاني s1 وs2، والمعدّل السنويّ MA، ومجموع الأرصدة المحصّل عليها Crd.

الحلّ

الخوارزم	لغة C

algorithme absolue
var s1, s2, MA:réel
Crd :entier
début
écrire("ادخل معدل السداسي الأول والثاني")
lire(s1,s2)
écrire("ادخل المعدل السنوي")
lire(MA)
écrire("ادخل مجموع الارصدة")
lire(Crd)
Si s1>=10 et s2>=10 **Alors**
écrire("ناجح بدون تعويض")
sinon
Si MA>=10 **Alors**
écrire("ناجح بالتعويض")
sinon
Si Crd>=45 **Alors**
écrire("ناجح بديون")
sinon
écrire("راسب")
FinSi
FinSi
FinSi
fin
	#include <stdio.h>
	int main(){
	float **s1**, s2, MA ;
	int Crd ;
	printf("ادخل معدل السداسي الأول والثاني\n") ;
	scanf("%f%f", &s1, &s2) ;
	printf("ادخل المعدل السنوي\n") ;
	scanf("%f", &MA) ;
	printf("ادخل مجموع الارصدة\n") ;
	scanf("%d", &Crd) ;
	if (s1>=10 && s2>=10 **)**
	printf("ناجح بدون تعويض") ;
	else if (MA>=10 **)**
	printf("ناجح بتعويض") ;
	else if (Crd>=45 **)**
	printf("ناجح بديون") ;
	else
	printf("راسب") ;
	}

4. التعليمة الشرطيّة للاختيارات المتعدّدة (التعليمة الإنتقائيّة) selon

لاختيار إجراء من بين اختيارين، نستعمل التعليمة si. لكن، في وجود عدّة اختيارات (أكثر من 2)، يمكن استعمال si المتداخلة، وذلك بتقسيم الاختيارات إلى اختيارين، هما: الاختيار الأول، وباقي الاختيارات، في كل مرة. ولكن، سينجم عنها si متداخلة بعدد الاختيارات، وهذا ما سيجعل البرنامج صعب القراءة.

الاختبار selon (حسب): هو حالة خاصّة لتعليمة si sinon المتداخلة. حيث تسمح بتحديد الكتلة المراد تنفيذها وفقا لقيمة المتغيّر. يتمّ استخدامها عندما يكون لدينا عدّة مخرجات ، ويتمّ اختبار الشرط عدّة مرّات ، دائما باستخدام نفس المتغيّر. فباستعمال selon يمكننا جعل البرنامج أكثر مقروئيّة. وتتكون من:

- التعبير expression: الذي سيتمّ اختباره، ويكون من نوع عدد صحيح او رموز (او منطقي وهنا من الأفضل استعمال si sinon). وفي العادة، تكون متغيرا واحدا. مثل: age.
- القيم المراد اختبارها مع كتلة التعليمات الخاصة بكل قيمة.

- كتلة تعليمات اختيارية في حالة عدم وجود أي قيمة توافق القيمة الحالية للمتغيّر (او العبارة).

4.1. الصيغة

الخوارزم	لغة C
selon expression **faire** **case** val_1 :1 كتلة التعليمات **case** val_2 :2 كتلة التعليمات … **case** val_n :n كتلة التعليمات **sinon** كتلة التعليمات اخرى **FinSelon** بقية التعليمات	**switch (expression)** { **case** val_1 : 1 كتلة التعليمات **break ;** … **case** val_n : n كتلة التعليمات **break ;** **default:** كتلة التعليمات اخرى **}** بقية التعليمات

حيث أنّ الكلمات **selon**، **faire**، **sinon**، **case** او **cas** و**FinSelon** أو **FSelon** هي كلمات محجوزة في الخوارزم.

كذلك الحال بالنسبة لـ **switch**، **case** و**default** في C.

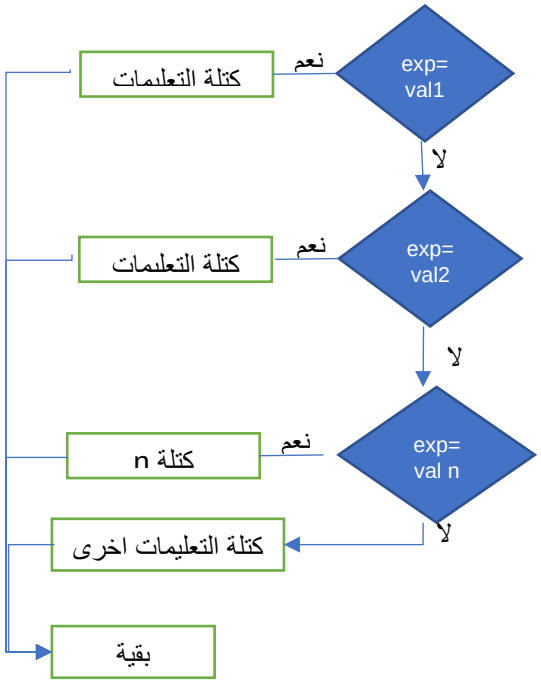
- expression: هي عبارة، يتمّ حسابها للحصول على قيمة من نوع عدد صحيح، أو رموز. وفي العادة، تكون عبارة عن متغير. وتكون في الخوارزم بين كلمة selon وfaire، بينما في C، فتكون بين قوسين ().
- val_1، ...، val_n: عبارة عن قيمة او ثابت من نفس نوع expression.
- كتلة التعليمات: وهي تعليمة او عدة تعليمات يتم تنفيذها في حالة تطابق قيمة expression مع val_i.

ملاحظة: تُستعمل selon بدل si المتداخلة إذا كنّا سنختبر عبارة **واحدة** أو متغيرا، من نوع عدد **صحيح** او **رموز**، عدّة مرات مع قيم **ثابتة**.

4.2. القواعد الخاصة بـ switch

- الحاضنتين {} الخاصة بـ **switch** والقوسين هي ضروريّة، ولا يمكن حذفها.
- يجب أن تكون كل قيمة val_i مختلفة عن الأخرى. على سبيل المثال ، من غير القانونيّ كتابة case 1 مرّتين.
- يمكن وضع case val_i بأيّ ترتيب. ومع ذلك ، ينصح بوضعها بترتيب تصاعديّ. ممّا يزيد من مقروئيّة البرنامج.
- كتلة التعليمات يمكن أن تكون بأيّ عدد من التعليمات، ومن أيّ نوع.
- التعليمة ; break اختياريّة. وتستعمل لإنهاء switch مباشرة، حيث تنقل تدفق البرنامج إلى خارج switch.
- الحالة default اختياريّة. إذا لم تتمّ مطابقة أيّ الحالات case val_i، فسيتمّ نقل سياق التنفيذ إلى الكتلة default. ويجب أن تكون هي الحالة الأخيرة.

4.3. الهيكل التنظيمي algorigramme



4.4. التنفيذ

تتمّ عمليّة تنفيذ التعليمة selon بحساب عبارة expression، ثمّ تذهب إلى القيمة التي تساويها من بين val_i، وينفّذ كتلة التعليمات الخاصّة بها، ثمّ يتمّ تنفيذ بقيّة تعليمات البرنامج. وفي حالة، لا توجد أيّ قيمة تساويها فإنّه ينفّذ كتلة التعليمات الخاصّة بـ sinon إن وجدت، ثمّ يتمّ تنفيذ بقيّة تعليمات البرنامج.

ويختلف تنفيذ switch في C عن selon قليلا، إذ، بعد تنفيذ كتلة التعليمات الخاصّة بـ val_i، ولم يصادف التنفيذ التعليمة break ;، فسيُواصل في تنفيذ كتلة التعليمات التي تليها، إلى أن يصادف التعليمة break ; ليتحوّل التنفيذ إلى بقيّة التعليمات خارج switch.

ولكي تكون switch مكافئة لـ selon، يجب إضافة break ; في نهاية كلّ كتلة.

إذا كان هناك قيمتين أو أكثر لهما نفس كتلة التعليمات، فيمكن في الخوارزم استعمال الفاصلة. بينما في C، فنترك القيمة الأولى بدون تعليمات ولا break ;. فلو فرضنا أنّ القيمة 7 و9 لهما نفس المعالجة فنكتب:

الخوارزم	لغة C

`case 7 ,9 : كتلة التعليمات`	`case 7 :`
	`case 9 :`
	`  كتلة التعليمات`
	`  break ;`

4.5. مثال

اكتب البرنامج الذي يقرأ عددا صحيحا أقل من 10، ثمّ يظهر على الشاشة هذا العدد بالإنجليزيّة.

الخوارزم	لغة C	الشاشة

	`#include <stdio.h>`
	`int main(){`
	`int nb ;`
	`printf("ادخل عددا صحيحا\n") ;`
	`scanf("%d", &nb ) ;`
	`switch ( nb ) {`
	`case 0 : printf("zero") ;`
	`break ;`
	`case 1 : printf("one") ;`
	`break ;`
	`…`
	`case 9 : printf("nine") ;`
	`break ;`
	`default:`
	`printf("not treated") ;`
	`}`
	`return 0 ;`
	`}`
		`algorithme conversion`
		`var nb :entier`
		`début`
		`écrire("ادخل عددا صحيحا")`
		`lire(nb)`
		`Selon nb faire`
		`case 0 : écrire("zero")`
		`case 1 : écrire("one")`
		`case 2 : écrire("two")`
		`…`
		`case 9 : écrire("nine")`
		`sinon`
		`écrire("not treated")`
		`FinSelon`
		`fin`

5. الجمل التفرّعيّة (الربط) Branchement

هو عملية الانتقال بين تعليمات البرنامج التي ينفّذها المعالج ، حيث يقوم بعملية "القفز" إلى عنوان محدّد بدلا من الإستمرار في تنفيذ التعليمات بالتتابع. وهناك أربع تعليمات في C بإمكانها تغيير السريان التسلسليّ للبرنامج دون قيد أو شرط، وهي : break (توقّف)، goto (إذهب إلى)، continue (إستمر) وreturn (أرجِع).

5.1. التعليمة break

سبق أن رأيناها مع switch حيث تؤدّي إلى إنهاء التعليمة switch، ونقل سريان التدفّق إلى أول تعليمة بعد switch. وفي حالة تعليمات switch متداخلة، فهي تؤدّي إلى الخروج من switch التي هي تابعة لها مباشرة فقط. وتُستعمل أيضا للخروج من الحلقات (الدرس التالي). وفي هذه الحالة، تكون ; break، في العادة، داخل if.

مثال

```
switch (grade){
case 'A' :
case 'a' : printf("excellent\n") ;
       break ;
case 'b' : printf("good\n") ;
case 'c' : printf("you can do better\n") ;
       break ;
default : printf("try again\n") ;
}
```

- إذا كانت grade تحوي الحرف a أو A، فستظهر excellent
- إذا كانت تحوي b، فستظهر good وyou can do better
- بينما إذا كانت تحتوي على الحرف c، فستظهر you can do better فقط
- إذا كانت تحوي على أيّ حرف اخر، فستظهر try again

5.2. التعليمة goto

تحوّل تنفيذ البرنامج إلى التعليمة التي تمّت تسميتها. ويُدعى هذا الاسم بالإنجليزيّة بـ label. وتتمّ تسميّة أيّ تعليمة بوضع معرّف صحيح ونقطتين قبلها.

صيغة التسمية label

```
label : تعليمة ;
```

حيث `label` (التسمية) هو أيّ معرّف صحيح. مثل:

```
ici : printf("zero") ;
```

وللانتقال إلى تلك التعليمة من أيّ مكان، نستعمل الصيغة التالية:

```
goto label ;
```

حيث `label` هو اسم التعليمة. مثل:

للذهاب إلى التعليمة `ici` من أيّ مكان، نستعمل:

```
goto ici ;
```

ملاحظة:

- تُعتبر : case val_i و: default طريقة تسمية خاصّة تُستعمل داخل switch.
- يمكن استعمال goto لتكرار التعليمات دون الحاجة للحلقات.
- ينصح بعدم استعمال goto والتسميات، لأنّه من الصعب على العقل البشري التكيّف معها. فالبرنامج الذي يستعمل goto كثيرا، من الصعب فهمه وصيانته.

مثال

```
again :
…
goto again ;
```

5.3. التعليمة continue

وتُستعمل مع الحلقات ، وتسمح بتحويل التدفّق إلى نهاية الحلقة، والمرور مباشرة إلى التكرار الموالي، دون إتمام تعليمات الحلقة. وتكون، في العادة، داخل if.

الصيغة

```
continue ;
```

5.4. التعليمة return

وتُستخدم للخروج من الدوال (الفصل الثاني) وارجاع النتيجة

الصيغة

```
return expression ;
```

مثال

```
return 0 ;
```

كالتي تُستعمل في نهاية الدالّة ()main.