

TP Structure des ordinateurs et applications

Corrigé de la série de TP N°4

Rappel :

Structures de contrôle conditionnel

Ces structures sont utilisées pour déterminer l'exécution d'un bloc d'instructions : soit ce bloc est exécuté, soit il ne l'est pas. Elles servent également à choisir entre l'exécution de deux blocs différents. Nous avons deux types de structures conditionnelles :

1. Test alternatif simple

Un test simple contient un seul bloc d'instructions. Selon une condition (expression logique), on décide si le bloc d'instructions est exécuté ou non. Si la condition est vraie, on exécute le bloc, sinon on ne l'exécute pas.

La syntaxe d'un test alternatif simple est la suivante :

Algorithme	Langage C
<pre>si <Condition> alors <bloc_instructions_si> ; finsi ;</pre>	<pre>if (Condition) { <bloc_instructions_if> ; }</pre>

Remarque : Dans le langage C, un bloc est délimité par deux accolades { et }. Si le bloc contient une seule instruction, les accolades { et } sont facultatives (on peut les enlever).

2. Test alternatif double

Un test double contient deux blocs d'instructions : on est amené à choisir entre le premier bloc ou le second. Cette décision est réalisée sur une condition (expression logique ou booléenne) qui peut être vraie ou fausse. Si la condition est vraie on exécute le premier bloc, sinon on exécute le second.

La syntaxe d'un test alternatif simple est la suivante :

Algorithme	Langage C
<pre>si <Condition> alors <bloc_instructions_si> sinon <bloc_instructions_sinon> ; finsi ;</pre>	<pre>if (Condition) { <bloc_instructions_if> ; } else { <bloc_instructions_else> ; }</pre>

Remarques :

- En langage C, il ne faut pas mettre de point-virgule après la condition (erreur logique).
- Dans l'exemple précédent, les accolades { } du **if** et du **else** peuvent être omises puisqu'il y a une seule instruction dans les deux blocs.

Nous avons aussi, les **structures conditionnelles doubles et imbriquées** :

Un test double et imbriqué, tout comme un test double, contient deux blocs instructions avec au moins un des deux blocs (bloc Si et/ou bloc Sinon) est composé d'une instruction de condition simple ou double. Donc un test double et imbriqué contient au moins trois blocs d'instructions avec au moins deux conditions.

La syntaxe d'un test alternatif double imbriqué avec trois blocs d'instructions est :

Algorithme	Langage C
<pre>si <Condition> alors <bloc_instructions_si> sinon si <Condition> alors <bloc_instructions_si> sinon <bloc_instructions_sinon> ; finsi ; finsi ;</pre>	<pre>if (Condition) { <bloc_instructions_if> ; } else if (Condition) { <bloc_instructions_if> ; } else { <bloc_instructions_else> ; }</pre>

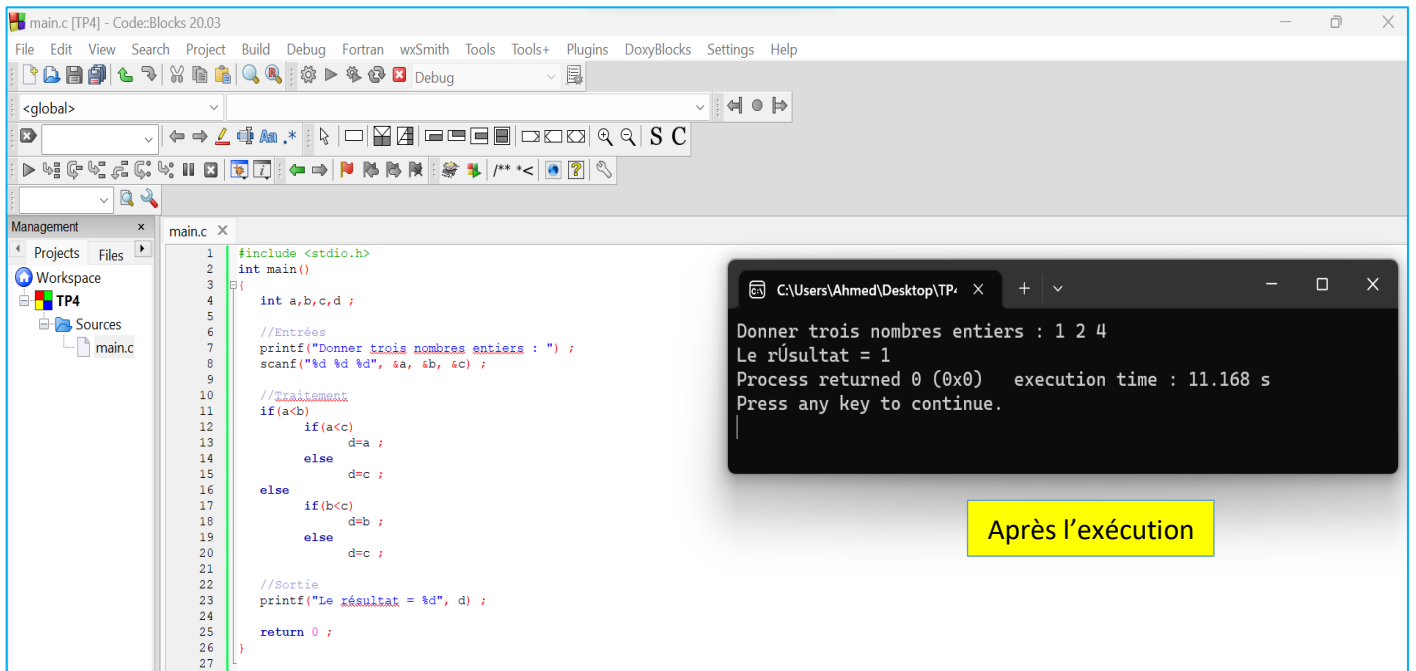
Exercice N°01 : (Algorithme → Programme en langage C)

1) Traduire l'algorithme en un programme en langage C

Algorithme	Programme C
<p>Algorithme exo1 ;</p> <p>Variables a, b, c, d : entier ;</p> <p>Début</p> <p><i>//Entrées</i></p> <p>Écrire ("Donner trois nombres entiers : ") ; Lire (a,b,c) ;</p> <p><i>//Traitement</i></p> <p>Si (a<b) alors Si (a<c) alors d ← a Sinon d ← c ; Fin-Si</p> <p>Sinon Si (b<c) alors d ← b Sinon d ← c ; Fin-Si</p> <p>Fin-Si</p> <p><i>//Sortie</i></p> <p>Écrire ("Le résultat = ", d) ;</p> <p>Fin.</p>	<pre>#include <stdio.h> int main() { int a,b,c,d ; <i>//Entrées</i> printf("Donner trois nombres entiers : ") ; scanf("%d %d %d", &a, &b, &c) ; <i>//Traitement</i> if(a<b) if(a<c) d=a ; else d=c ; else if(b<c) d=b ; else d=c ; <i>//Sortie</i> printf("Le résultat = %d", d) ; return 0 ; }</pre>

2) Compiler et exécuter le programme pour les valeurs suivantes :

➤ a=1, b=2, c=4



The screenshot shows the Code::Blocks IDE with a C program named 'main.c'. The code is as follows:

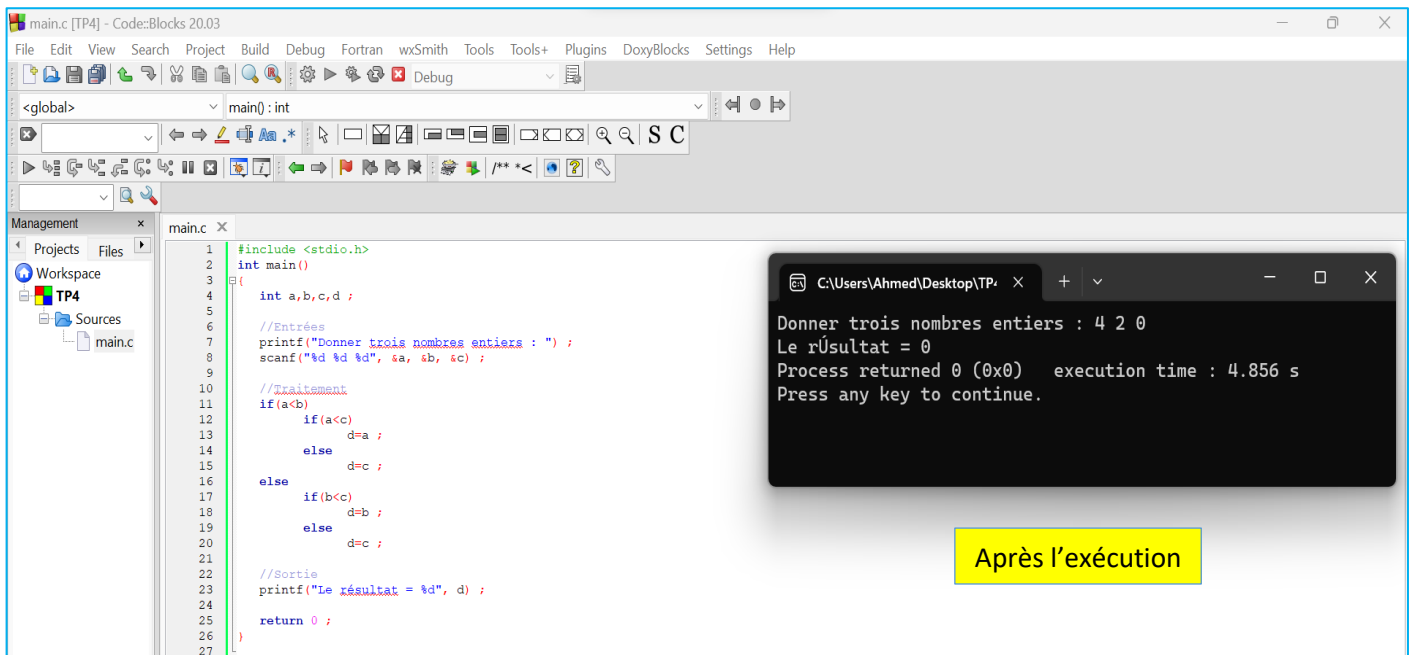
```
1 #include <stdio.h>
2 int main()
3 {
4     int a,b,c,d ;
5
6     //Entrées
7     printf("Donner trois nombres entiers : ") ;
8     scanf("%d %d %d", &a, &b, &c) ;
9
10    //Traitement
11    if(a<b)
12        if(a<c)
13            d=a ;
14        else
15            d=c ;
16    else
17        if(b<c)
18            d=b ;
19        else
20            d=c ;
21
22    //Sortie
23    printf("Le résultat = %d", d) ;
24
25    return 0 ;
26 }
27
```

The output window shows the following text:

```
C:\Users\Ahmed\Desktop\TP: x + v
Donner trois nombres entiers : 1 2 4
Le résultat = 1
Process returned 0 (0x0) execution time : 11.168 s
Press any key to continue.
```

Après l'exécution

➤ a=4, b=2, c=0



The screenshot shows the Code::Blocks IDE with the same C program as above. The code is identical:

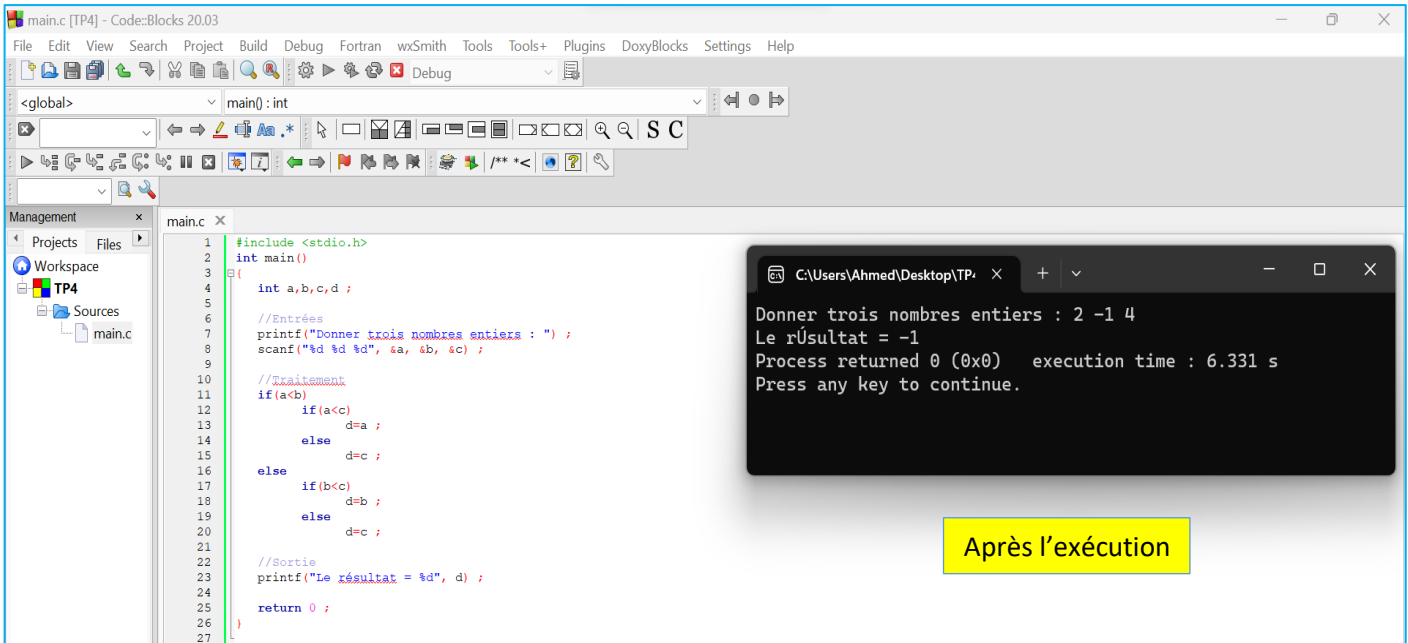
```
1 #include <stdio.h>
2 int main()
3 {
4     int a,b,c,d ;
5
6     //Entrées
7     printf("Donner trois nombres entiers : ") ;
8     scanf("%d %d %d", &a, &b, &c) ;
9
10    //Traitement
11    if(a<b)
12        if(a<c)
13            d=a ;
14        else
15            d=c ;
16    else
17        if(b<c)
18            d=b ;
19        else
20            d=c ;
21
22    //Sortie
23    printf("Le résultat = %d", d) ;
24
25    return 0 ;
26 }
27
```

The output window shows the following text:

```
C:\Users\Ahmed\Desktop\TP: x + v
Donner trois nombres entiers : 4 2 0
Le résultat = 0
Process returned 0 (0x0) execution time : 4.856 s
Press any key to continue.
```

Après l'exécution

➤ a=2, b= -1, c=4



Après l'exécution

3) Dédire ce que fait cet algorithme ?

Le programme donne le plus petit des trois nombres entiers

4-1) Déroulement du programme pour a=1, b=2 et c=4

Instructions	Variables				Affichage
	a	b	c	d	
Écrire ("Donner trois nombres entiers : ");	/	/	/	/	Donner trois nombres entiers :
Lire (a,b,c) ;	1	2	4	/	
Si (a<b) alors 1 < 2 True ⇒ On exécute le bloc Si Si (a<c) alors 1 < 4 True ⇒ On exécute le bloc Si d ← a d ← 1 Fin-Si	1	2	4	1	
Écrire ("Le résultat = ", d) ;	1	2	4	1	Le résultat = 1

4-2) Déroulement du programme pour a=4, b=2 et c=0

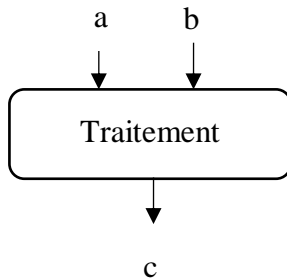
Instructions	Variables				Affichage
	a	b	c	d	
Écrire ("Donner trois nombres entiers : ");	/	/	/	/	Donner trois nombres entiers :
Lire (a,b,c);	4	2	0	/	
<p>Si (a<b) alors</p> <p>4 < 2 False</p> <p>⇒ On n'exécute pas le bloc Si (on passe au bloc Sinon)</p> <p>Si (b<c) alors</p> <p>4 < 0 False</p> <p>⇒ On n'exécute pas le bloc Si (on passe au bloc Sinon)</p> <p>d ← c;</p> <p>d ← 0</p> <p>Fin-Si</p>	4	2	0	0	
Écrire ("Le résultat = ", d);	4	2	0	0	Le résultat = 0

4-3) Déroulement du programme pour a=2, b=-1 et c=4

Instructions	Variables				Affichage
	a	b	c	d	
Écrire ("Donner trois nombres entiers : ");	/	/	/	/	Donner trois nombres entiers :
Lire (a,b,c);	2	-1	4	/	
<p>Si (a<b) alors</p> <p>2 < -1 False</p> <p>⇒ On n'exécute pas le bloc Si (on passe au bloc Sinon)</p> <p>Si (b<c) alors</p> <p>-1 < 4 True</p> <p>⇒ On n'exécute le bloc Si</p> <p>d ← b;</p> <p>d ← -1</p> <p>Fin-Si</p>	2	-1	4	-1	
Écrire ("Le résultat = ", d);	2	-1	4	-1	Le résultat = -1

Exercice N°02 :

Le schéma entrées, traitement et sorties.



Entrées :

Lire (a,b) ;

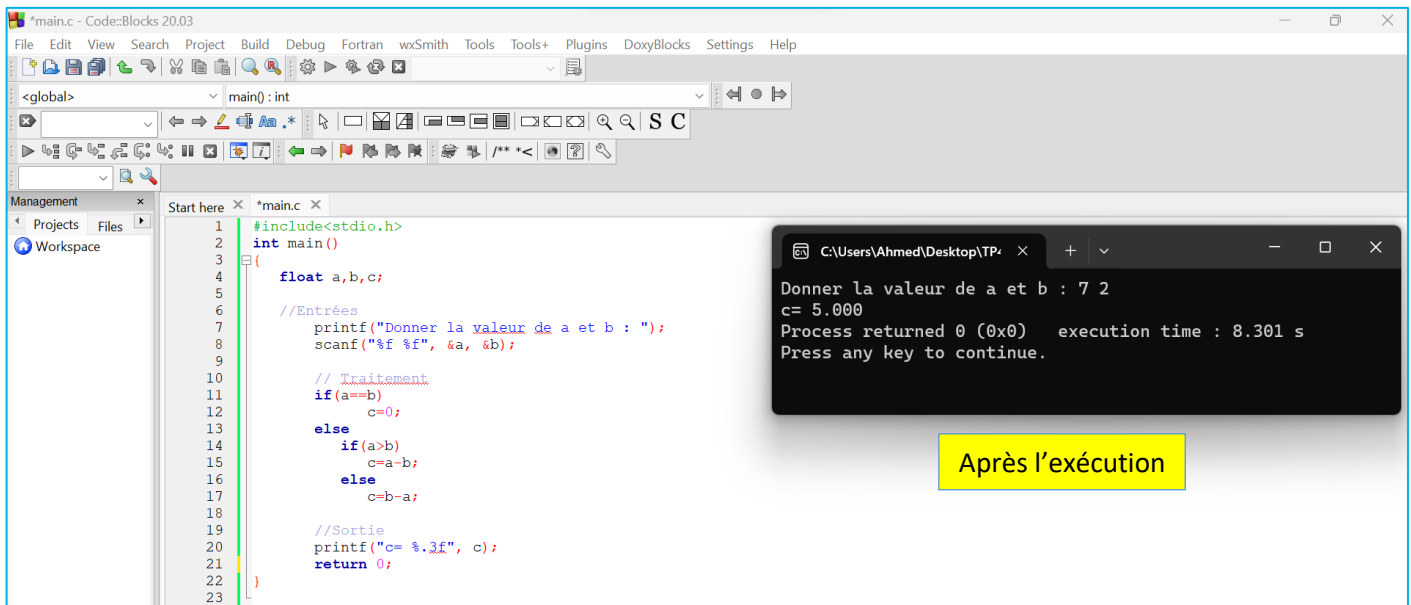
Traitements :

```
Si a=b alors
    c ← 0;
Sinon
    Si a>b alors
        c ← a-b;
    Sinon
        c ← b-a;
Fin-Si
Fin-Si
```

Sortie :

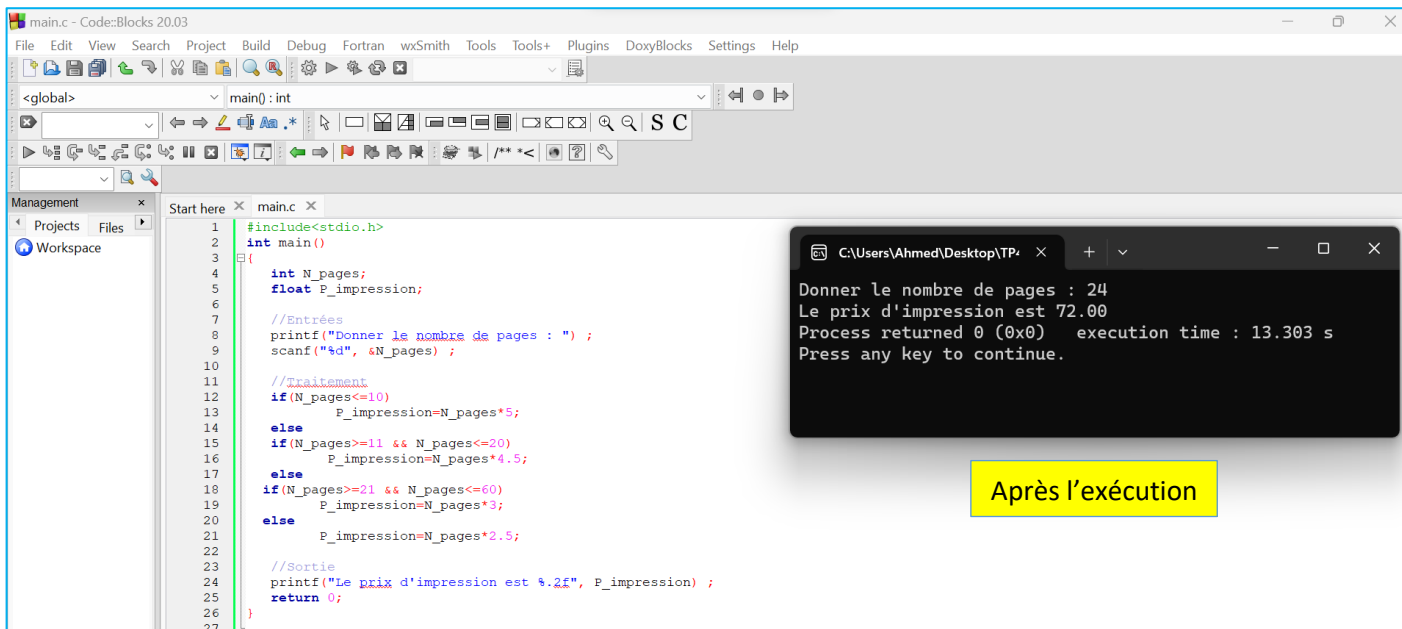
Écrire (c) ;

Algorithme	Programme C
<pre>Algorithme exo2; Variables a, b, c : réel; Début //Entrées Ecrire("Donner la valeur de a et b : "); Lire (a,b); //Traitement Si (a=b) alors c ← 0; Sinon Si (a>b) alors c ← a-b; Sinon c ← b-a; Fin-Si Fin-Si //Sortie Ecrire ("c = ", c) Fin.</pre>	<pre>#include<stdio.h> int main() { float a,b,c; //Entrées printf("Donner la valeur de a et b : "); scanf("%f %f", &a, &b); // Traitement if(a==b) c=0; else if(a>b) c=a-b; else c=b-a; //Sortie printf("c= %.3f", c); return 0; }</pre>



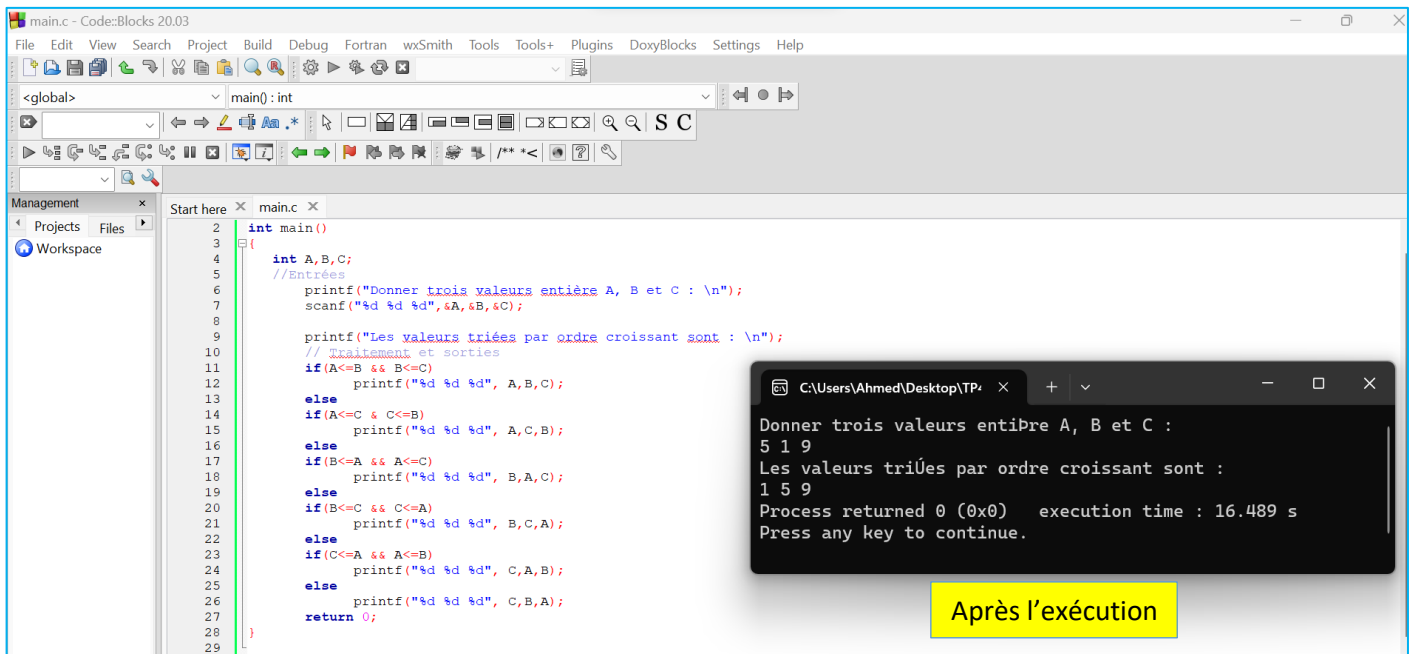
Exercice N°03 :

Algorithme	Programme C
<p>Algorithme Prix_impression;</p> <p>Variables</p> <p>N_pages : entier; P_impression : réel;</p> <p>Début</p> <p><i>//Entrées</i></p> <p>Écrire ("Donner le nombre de pages : "); Lire (N_pages);</p> <p><i>//Traitement</i></p> <p>Si (N_pages <= 10) alors P_impression ← N_pages*5 ;</p> <p>Sinon</p> <p>Si (N_pages >= 11 ET N_pages <= 20) alors P_impression ← N_pages*4.5 ;</p> <p>Sinon</p> <p>Si (N_pages >= 21 ET N_pages <= 60) alors P_impression ← N_pages*3 ;</p> <p>Sinon P_impression ← N_pages*2.5 ;</p> <p>Fin-Si ; Fin-Si ; Fin-Si ;</p> <p><i>//Sortie</i></p> <p>Écrire("Le prix d'impression est ", P_impression);</p> <p>Fin.</p>	<pre> #include<stdio.h> int main() { int N_pages; float P_impression; //Entrées printf("Donner le nombre de pages : "); scanf("%d", &N_pages); //Traitement if(N_pages<=10) P_impression=N_pages*5; else if(N_pages>=11 && N_pages<=20) P_impression=N_pages*4.5; else if(N_pages>=21 && N_pages<=60) P_impression=N_pages*3; else P_impression=N_pages*2.5; //Sortie printf("Le prix d'impression est %.2f", P_impression); return 0; } </pre>



Exercice N°04 :

Algorithme	Programme C
<pre>Algorithme ordre_croissant; Variables A, B, C : entier; Début //Entrées Écrire("Donner trois valeurs entière A, B et C : "); Read(A, B, C); //Traitement & Sorties Écrire("Les valeurs triées par ordre croissant sont : "); Si (A <= B ET B <= C) alors Écrire(A, B, C); Sinon Si (A <= C ET C <= B) alors Écrire (A, C, B); Sinon Si (B <= A ET A <= C) alors Écrire(B, A, C) Sinon Si (B <= C ET C<=A) alors Écrire(B, C, A) Sinon Si (C <= A ET A<=B) alors Écrire(C, A, B); Sinon Écrire(C, B, A); Fin-Si ; Fin-Si ; Fin-Si ; Fin-Si ; Fin-Si ; Fin.</pre>	<pre>#include<stdio.h> int main() { int A,B,C; //Entrées printf("Donner trois valeurs entière A, B et C : \n"); scanf("%d %d %d",&A,&B,&C); // Traitement et sorties printf("Les valeurs triées par ordre croissant sont : \n"); if(A<=B && B<=C) printf("%d %d %d", A,B,C); else if(A<=C & C<=B) printf("%d %d %d", A,C,B); else if(B<=A && A<=C) printf("%d %d %d", B,A,C); else if(B<=C && C<=A) printf("%d %d %d", B,C,A); else if(C<=A && A<=B) printf("%d %d %d", C,A,B); else printf("%d %d %d", C,B,A); return 0; }</pre>



N.B : Il existe d'autres solutions qui permettent d'afficher trois valeurs numériques A, B et C avec ordre croissant.