Université A. Mira de Bejaia
Faculté de Technologie
Département de Technologie

TP Structure des ordinateurs et applications

Corrigé de la série de TP N°5

Rappel :

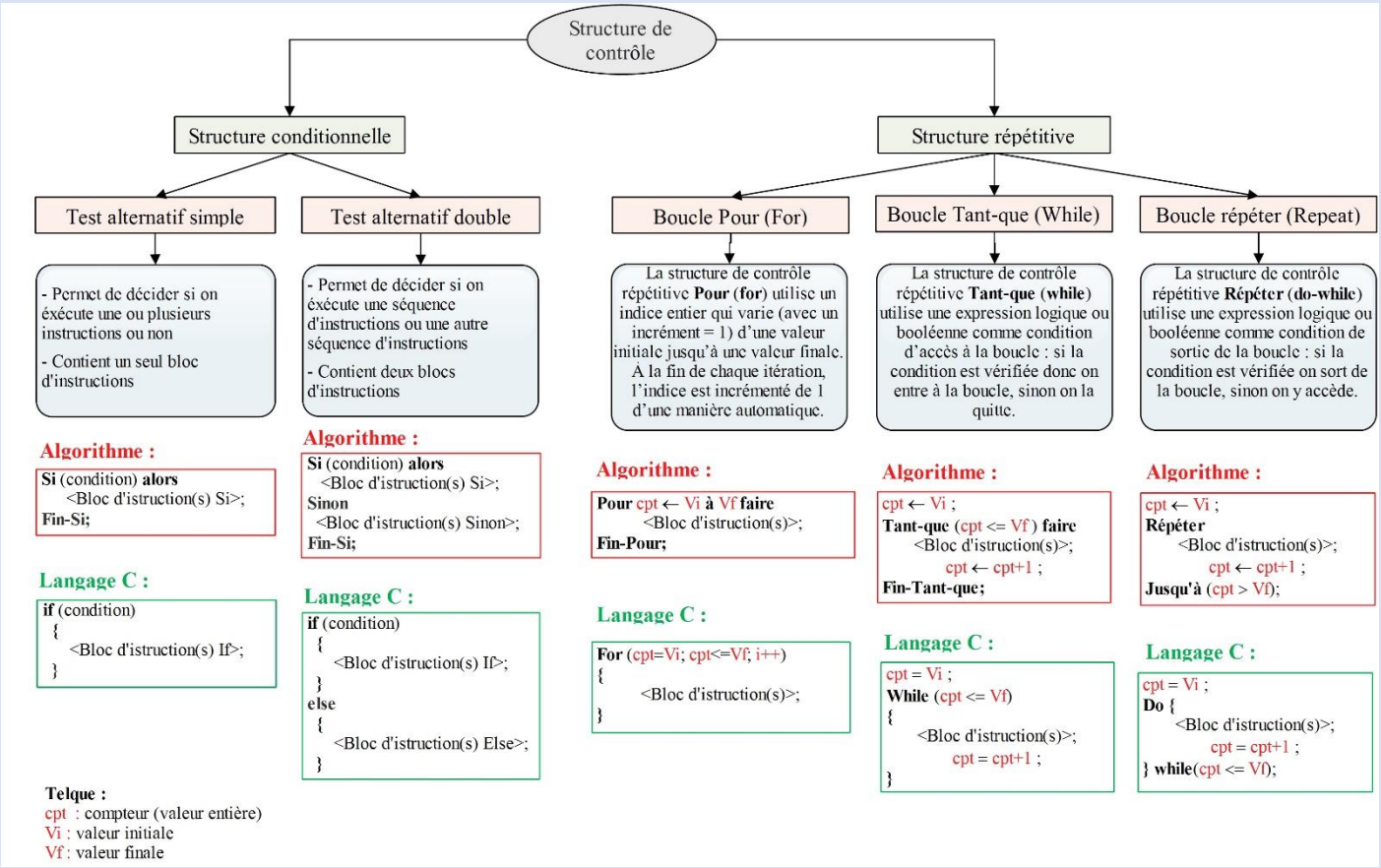
Structures de contrôle répétitives

Les structures répétitives nous permettent de répéter un traitement un nombre fini de fois. *Par exemple*, on veut afficher tous les nombres premiers entre 1 et N (N est un nombre entier positif donné).

Nous avons trois types de structures itératives (boucles) :

1. **Boucle pour (For)**
2. **Boucle tant-que (while)**
3. **Boucle répéter (do-while)**

Les syntaxes des trois boucles sont illustrées dans la figure ci-dessous :



La différence entre la boucle **répéter** et la boucle **tant-que** est :

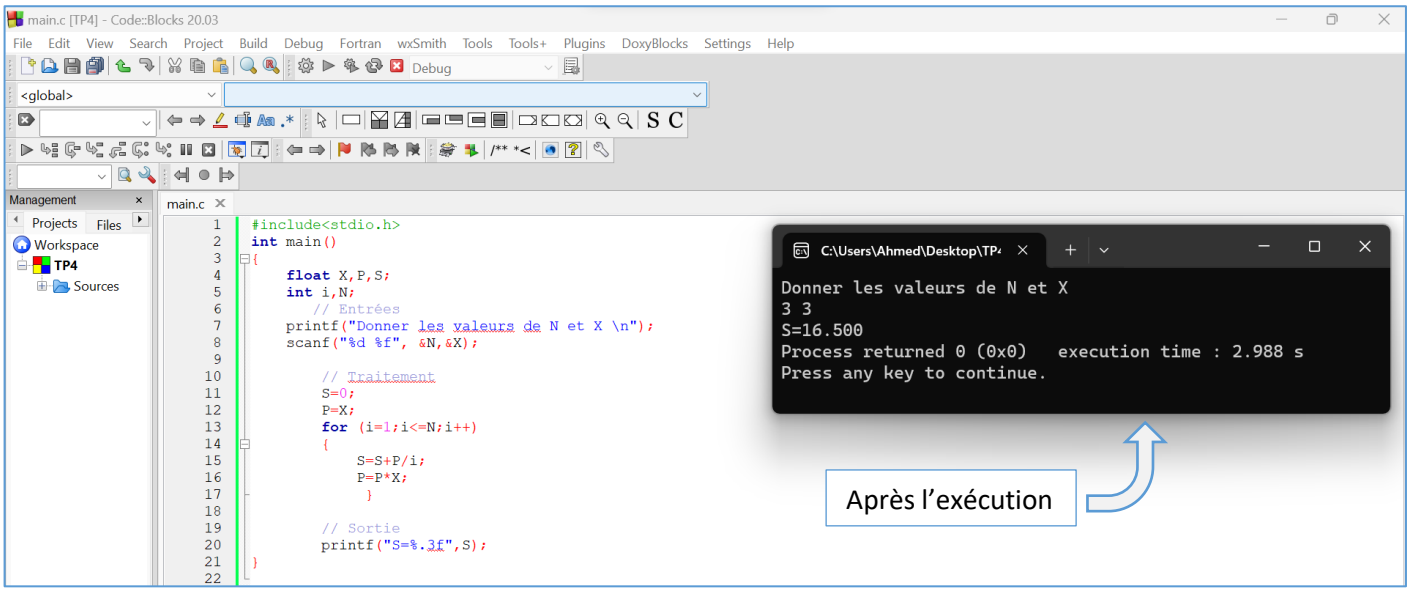
- La condition de **répéter** et toujours l'inverse de la condition **tant-que** : pour **répéter** c'est la condition de sortie de la boucle, et pour **tant-que** c'est la condition d'entrer.
- Le teste de la condition est à la fin de la boucle (la fin de l'itération) pour **répéter**. Par contre, il est au début de l'itération pour la boucle **tant-que**. C'est-à-dire, dans **tant-que** on teste la condition avant d'entrer à l'itération, et dans **répéter** on fait l'itération après on teste la condition.

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

1) Traduire l'algorithme en Programme C

Algorithme	Programme C
Algorithme Exo1 ; **Variables** X, P, S : **réel** ; i, N : **entier** ; **Début** // Entrées **Ecrire** ("Donner les valeurs de N et X : ") ; **Lire** (N, X) ; // Traitement S ← 0 ; P ← X ; **Pour** i ←1 **à** N **faire** S ← S+ P/i ; P ← P*X ; **Fin-Pour** // Sortie **Ecrire** ("S = ", S) ; **Fin.**	**#include<stdio.h>** **int main()** **{** **float** X,P,S; **int** i,N; // Entrées **printf**("Donner les valeurs de N et X \n"); **scanf**("%d %f", &N,&X); // Traitement S=0; P=X; **for** (i=1;i<=N;i++) { S=S+P/i; P=P*X; } // Sortie **printf**("S=%.3f",S); **}**

2) Compiler et exécuter le programme pour N=3, X=3.



3) Dérouler le programme pour N=3, X=3.

Instructions	Variables					Affichage
	N	X	i	P	S	
Lire (N,X);	3	3	/	/	/	/
S ← 0;	3	3	/	/	0	/
P ← X; P ← 3	3	3	/	3	0	/
Pour i ←1 S ← S+ P/i ; S ← 0+ 3/1 ; S ← 3 ; P ← P*X ; P ← 3*3 ; P ← 9 ;	3	3	1	**9**	**3**	
Pour i ← 2 S ← S+ P/i ; S ← 0+ 3/1+ 9/2 ; S ← **7.5** ; P ← P*X ; P ← 3*3*3 ; P ← 27 ;	3	3	1	**27**	**7.5**	
Pour i ← 3 S← S+ P/i ; S ← 0+ 3/1+ 9/2 + 27/3 ; S← **16.5**; P ← P*X ; P ← 3*3*3*3 ; P ← 81 ;	3	3	1	**81**	**16.5**	
Ecrire ('S = ', S) ;	3	3	1	**81**	**16.5**	S = **16.500**

4) Déduire l'expression générale du résultat S en fonction de X et N ?

Selon le déroulement ci-dessus, nous avons :

Pour i = 1, nous avons S = 3

Pour i = 2, nous avons S = $3 + \frac{9}{2} = 7.5$

Pour i = 3, nous avons S = $3 + \frac{9}{2} + \frac{27}{3} = 16.5$

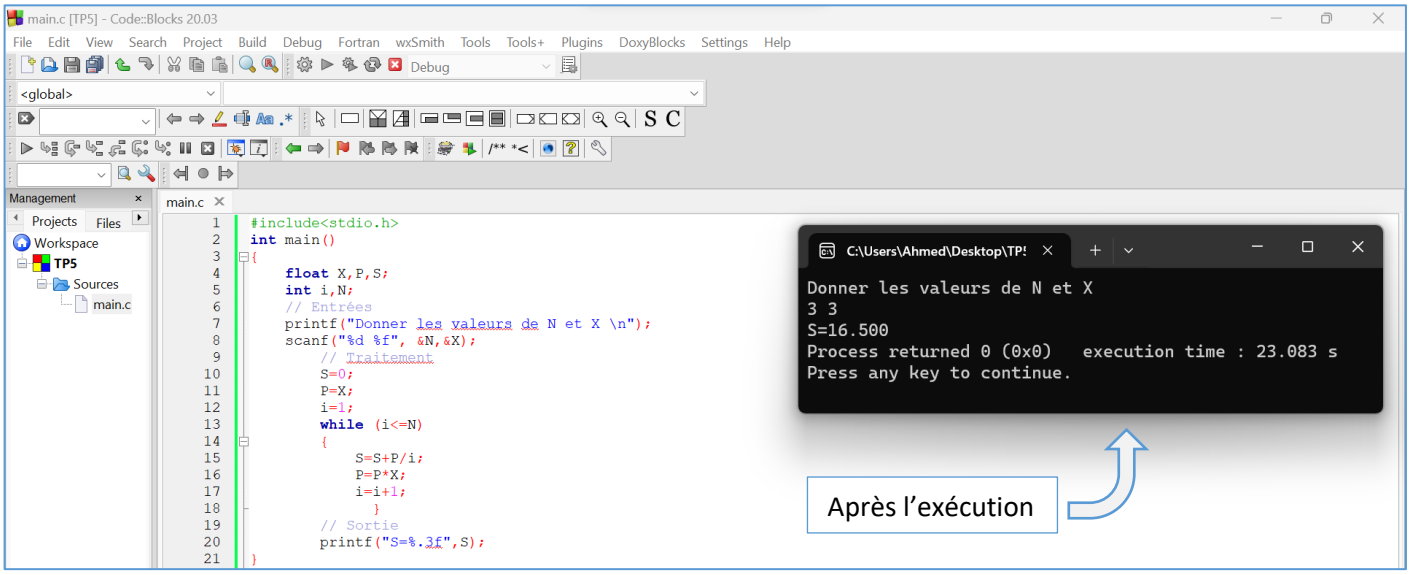
Pour i = N nous aurons : $S = X + \frac{X^2}{2} + \frac{X^3}{3} + \ldots + \frac{X^N}{N}$

On peut généraliser par la formule suivante :

$$S = \sum_{i=1}^{N} \frac{X^i}{i} \quad ou \quad S = X + \frac{X^2}{2} + \frac{X^3}{3} + \cdots + \frac{X^N}{N}$$

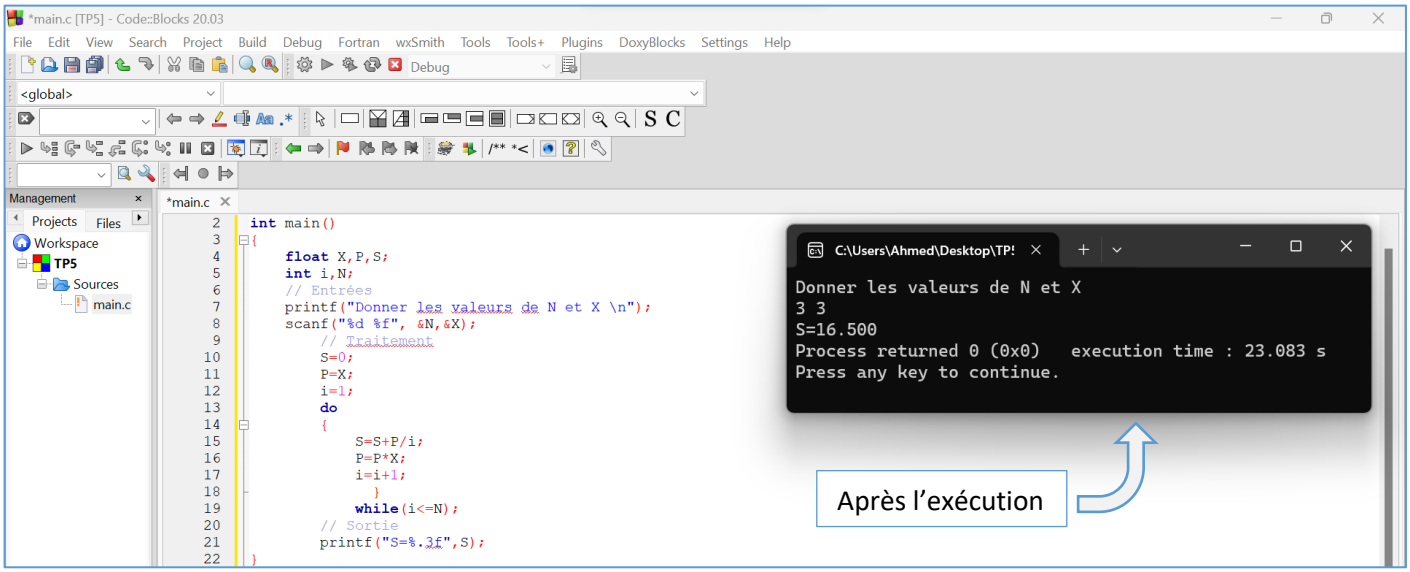
5) Réécrire l'algorithme en remplaçant la boucle *Pour* par la boucle *Tant-que*.

Algorithme	Programme C
Algorithme Exo1 ; **Variables** X, P, S : **réel** ; i, N : **entier** ; **Début** // Entrées **Ecrire** ("Donner les valeurs de N et X : ") ; **Lire** (N, X) ; // Traitement S ← 0 ; P ← X ; i ← 1 ; **Tant-que** i ≤ N **faire** S ← S+ P/i ; P ← P*X ; i ← i+1 ; **Fin Tant-que** // Sortie **Ecrire** ("S =", S) ; **Fin.**	**#include<stdio.h>** **int main()** **{** **float** X,P,S; **int** i,N; // Entrées **printf**("Donner les valeurs de N et X \n"); **scanf**("%d %f", &N,&X); // Traitement S=0; P=X; i=1; **while** (i<=N) { S=S+P/i; P=P*X; i=i+1; } // Sortie **printf**("S=%.3f",S); **}**



6) Réécrire l'algorithme en remplaçant la boucle *Pour* par la boucle *Répéter*.

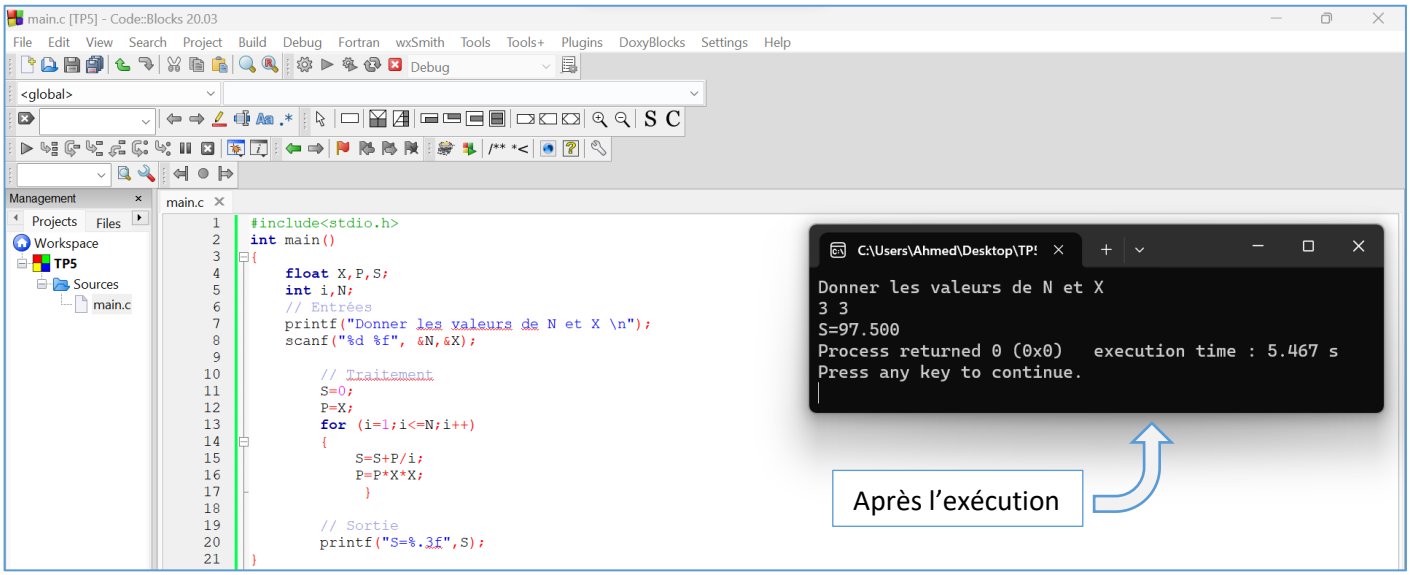
Algorithme	Programme C
Algorithme Exo1 ; **Variables** X, P, S : **réel** ; i, N : **entier** ; **Début** // Entrées **Ecrire** ("Donner les valeurs de N et X : ") ; **Lire** (N, X) ; // Traitement S ← 0 ; P ← X ; i ← 1 ; **Répéter** S ← S+ P/i ; P ← P*X ; i ← i+1 ; **Jusqu'a** i>N; // Sortie **Ecrire** ("S =", S) ; **Fin.**	**#include<stdio.h>** **int main()** **{** **float** X,P,S; **int** i,N; // Entrées **printf**("Donner les valeurs de N et X \n"); **scanf**("%d %f", &N,&X); // Traitement S=0; P=X; i=1; **do** **{** S=S+P/i; P=P*X; i=i+1; **}** **while**(i<=N); // Sortie **printf**("S=%.3f",S); **}**



7) Modifier l'algorithme pour calculer la somme S2 :

$$S2 = X + \frac{X^3}{2} + \frac{X^5}{3} + \frac{X^7}{4} + \cdots + N^{ème}\ terme$$

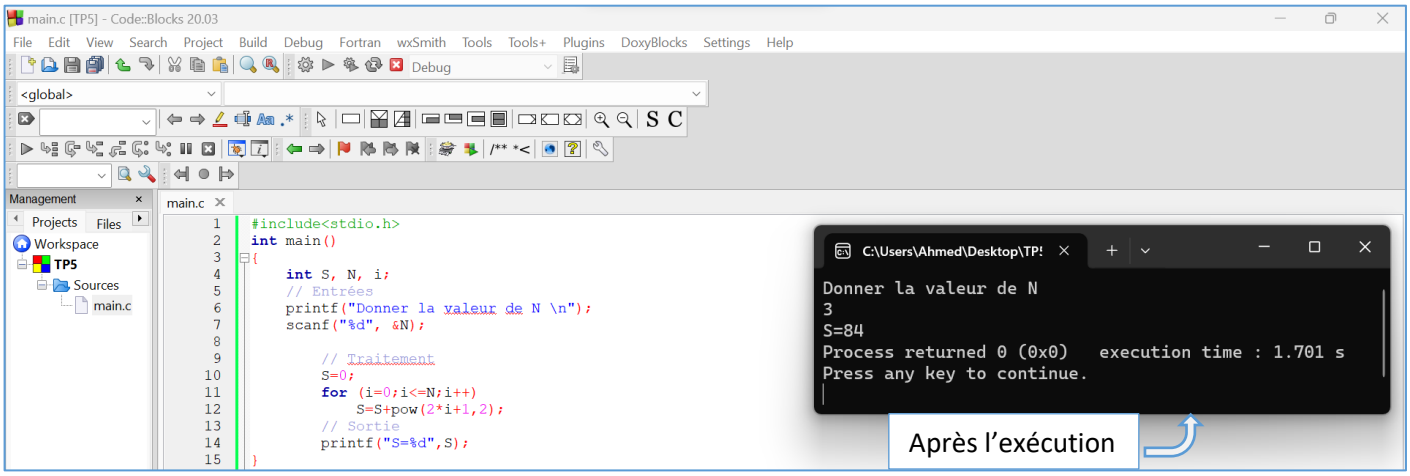
Algorithme	Programme C
Algorithme Exo2 ; **Variables** X, P, S : **réel** ; i, N : **entier** ; **Début** // Entrées **Ecrire** ("Donner les valeurs de N et X : ") ; **Lire** (N, X) ; // Traitement S ← 0 ; P ← X ; **Pour** i ←1 **à** N **faire** S ← S+ P/i ; P ← P*X*X ; **Fin-Pour;** // Sortie **Ecrire** ("S =", S) ; **Fin.**	**#include<stdio.h>** **int main()** **{** **float** X,P,S; **int** i,N; // Entrées **printf**("Donner les valeurs de N et X \n"); **scanf**("%d %f", &N,&X); // Traitement S=0; P=X; **for** (i=1;i<=N;i++) { S=S+P/i; P=P*X*X; } // Sortie **printf**("S=%.3f",S); **}**



Exercice N°02 : *(Enoncé du problème ➔ Algorithme ➔ Programme en langage C)*

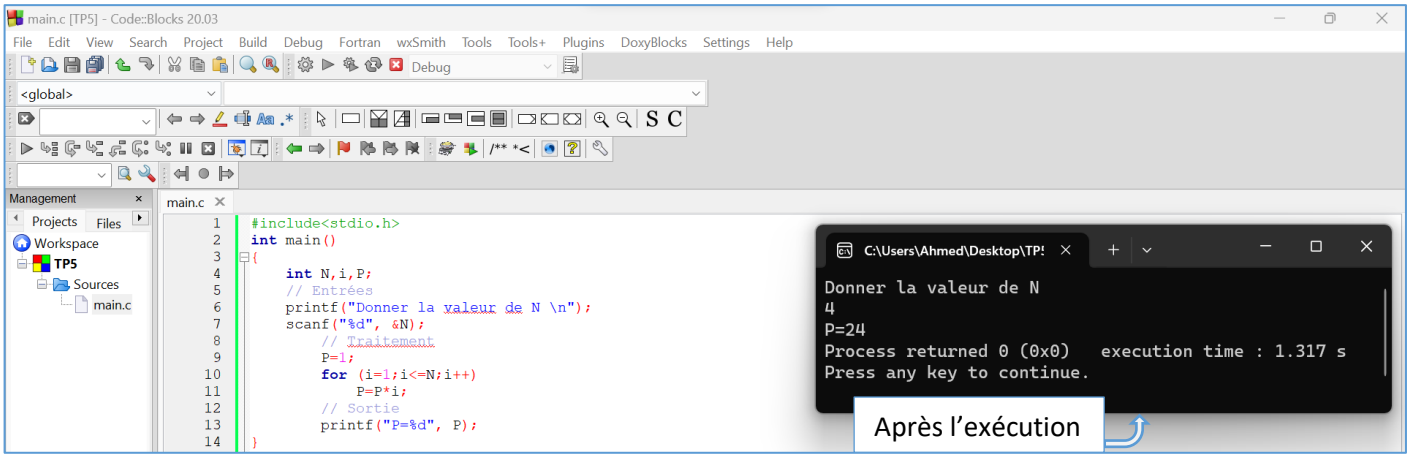
1) Calculer la somme $S = 1^2 + 3^2 + 5^2 + \cdots + (2 * N + 1)^2$

Algorithme	**Programme C**
Algorithme exo2_1; **variables** S, N, i: **entier**; **Début** // Entrées **Ecrire**("Donner la valeur de N : "); **Lire**(N); // Traitement S ← 0; **Pour** i ← 0 **à** N **faire** S ← S + (2*i+1)^2; **Fin-pour** // Sortie **Ecrire** ("S =", S); **Fin.**	**#include<stdio.h>** **int main()** **{** **int** S, N, i; // Entrées **printf**("Donner la valeur de N \n"); **scanf**("%d", &N); // Traitement S=0; **for** (i=0;i<=N;i++) S=S+pow(2*i+1,2); // S=S+(2*i+1)* (2*i+1); // Sortie **printf**("S=%d",S); **}**



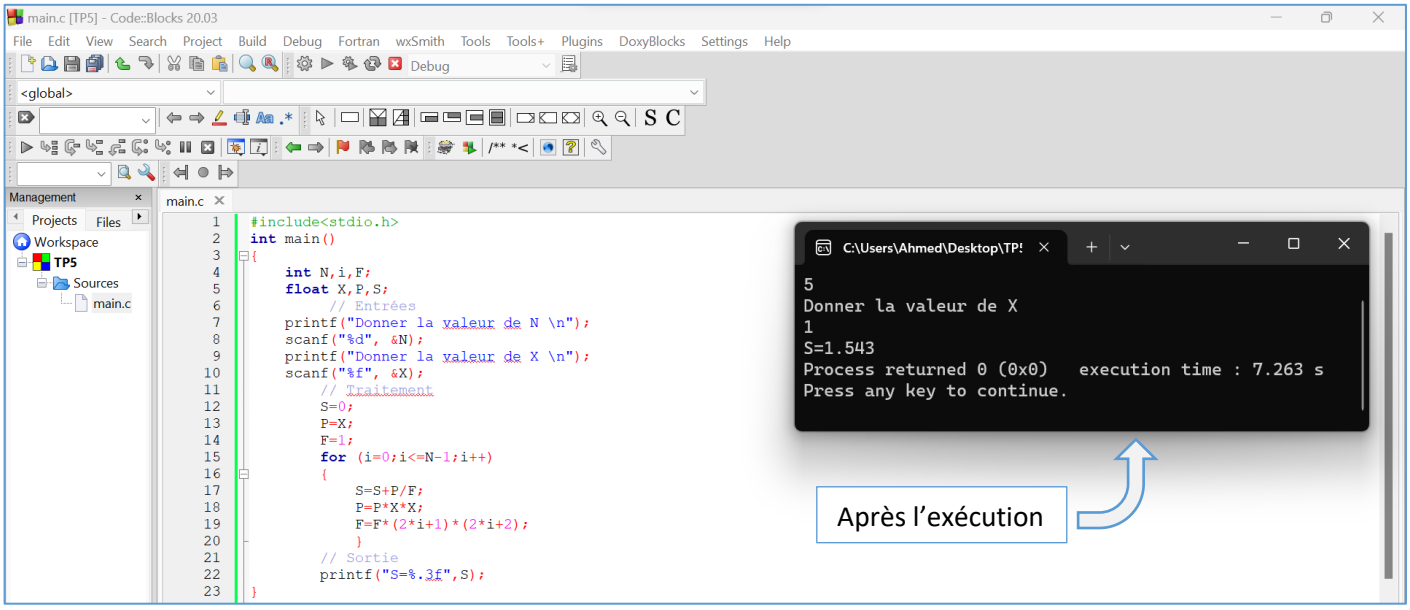
2) Calculer le produit $P = 1 \times 2 \times 3 \times ... \times N$

Algorithme	Programme C
Algorithme exo2_2; **Variables** N, i, P: **entier** ; **Début** // Entrées **Ecrire**("Donner la valeur de N : "); **Lire**(N); // Traitement P ← 1; **Pour** i ← 1 **à** N **faire** P ← P* i; **Fin-pour;** // Sortie **Ecrire**("P =", P); **Fin.**	**#include<stdio.h>** **int main()** **{** **int** N,i,P; // Entrées **printf**("Donner la valeur de N \n"); **scanf**("%d", &N); // Traitement P=1; **for** (i=1;i<=N;i++) P=P*i; // Sortie **printf**("P=%d", P); **}**



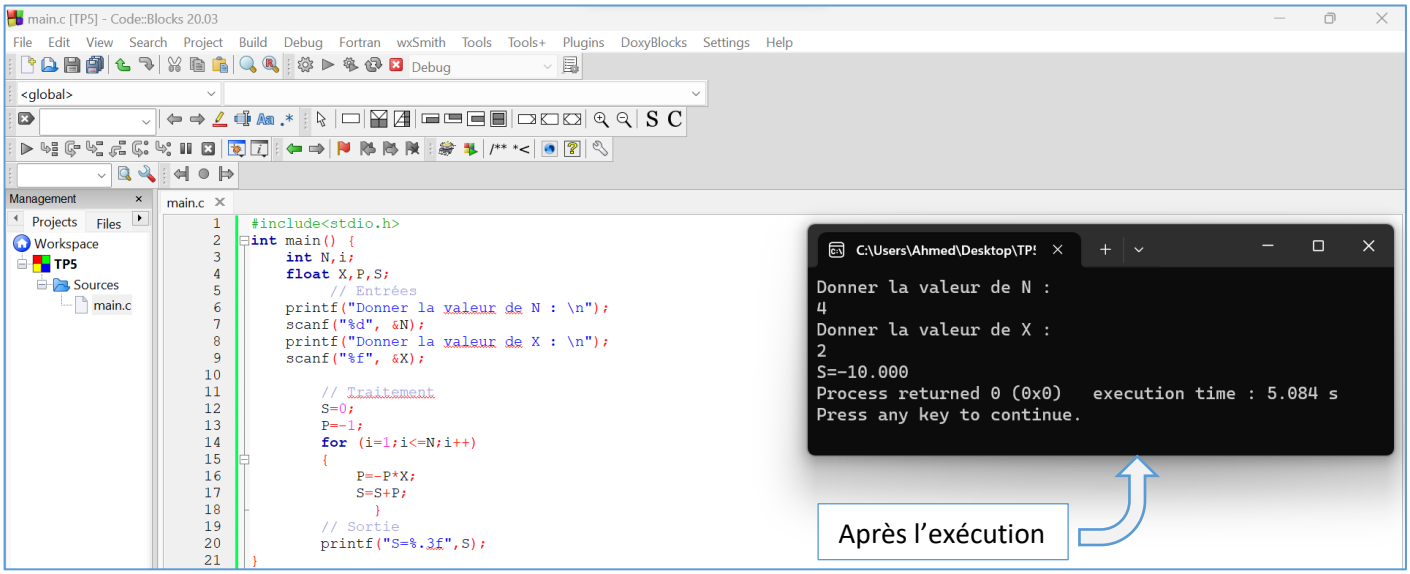
3) Calculer la somme $S = X + \frac{X^3}{2!} + \frac{X^5}{4!} + \frac{X^7}{6!} + \cdots$ ($N^{ème}$ terme)

Algorithme	**Programme C**
Algorithme exo2_3; **Variables** N, i, F : **entier**; X, P, S : **réel**; **Début** // Entrées **Ecrire**("Donner la valeur de N : "); **Lire**(N); **Ecrire**("Donner la valeur de X : "); **Lire**(X); // Traitement S ← 0; P ← X; F ← 1; **Pour** i ← 0 **à** (N-1) **faire** S ← S + P/F; P ← P*X*X; F ← F*(2*i+1)* (2*i+2); **Fin-Pour;** // Sortie **Ecrire**("S = ", S); **Fin.**	**#include<stdio.h>** **int main() {** **int** N,i,F; **float** X,P,S; // Entrées **printf**("Donner la valeur de N \n"); **scanf**("%d", &N); **printf**("Donner la valeur de X \n"); **scanf**("%f", &X); // Traitement S=0; P=X; F=1; **for** (i=0;i<=N-1;i++) { S=S+P/F; P=P*X*X; F=F*(2*i+1)*(2*i+2); } // Sortie **printf**("S=%.3f",S); **}**



4) Calculer la somme $S = X - X^2 + X^3 - \cdots \pm X^N$

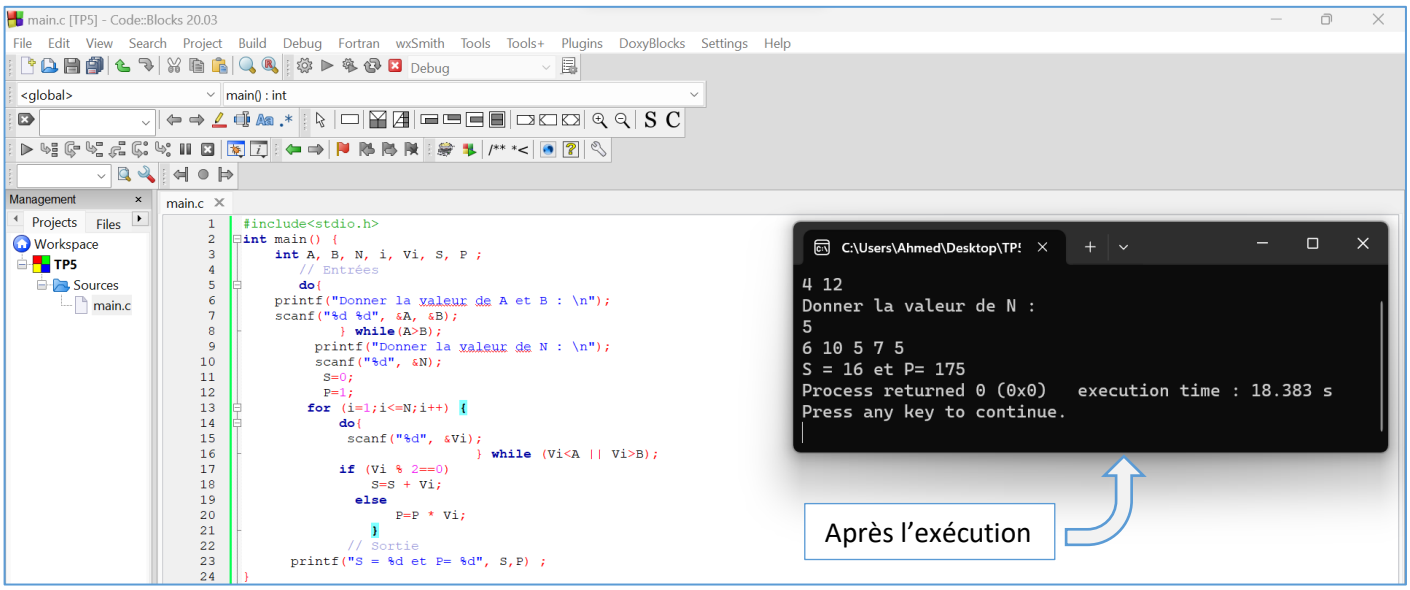
Algorithme	Programme C
Algorithme exo2_4; **Variables** N, i : **entier**; X, P, S : **réel**; **Début** // Entrées **Ecrire**("Donner la valeur de N : "); **Lire**(N); **Ecrire**("Donner la valeur de X : "); **Lire**(X); // Traitement S ← 0; P ← -1; **Pour** i ← 1 **à** N **faire** P ← -P*X; S ← S+P; **Fin-Pour;** // Sortie **Ecrire**("S= ", S); **Fin.**	**#include<stdio.h>** **int main() {** **int** N,i; **float** X,P,S; // Entrées **printf**("Donner la valeur de N : \n"); **scanf**("%d", &N); **printf**("Donner la valeur de X : \n"); **scanf**("%f", &X); // Traitement S=0; P=-1; **for** (i=1;i<=N;i++) { P=-P*X; S=S+P; } // Sortie **printf**("S=%.3f",S); **}**



Exercice N°03 :

Soit A et B deux entier tel-que A < B. Introduire N valeurs entières entre A et B, et réaliser la somme de valeurs pairs non-nuls et le produit des valeurs impaires.

Algorithme	Programme C
Algorithme Exo3; **Variables** A, B, N, i, Vi, S, P : **entier**; **Début** // Entrées **Répéter** **Ecrire**("Donner la valeur de A et B : ") ; **Lire**(A, B) ; **Jusqu'à** (A<B); **Ecrire**("Donner la valeur de N : ") ; **Lire**(N) ; S ← 0; P ← 1; **Pour** i ← 1 **à** N **faire** **Répéter** **Lire**(Vi) **Jusqu'à** (Vi>=A) **ET** (Vi<=B); **Si** Vi mod 2 = 0 **Alors** S ← S + Vi; **Sinon** P ← P * Vi; **Fin-Si;** **Fin-Pour;** // Sortie **Ecrire**("S = ", S, " P = ", P) ; **Fin.**	**#include<stdio.h>** **int main()** **{** **int** A, B, N, i, Vi, S, P ; // Entrées **do{** **printf**("Donner la valeur de A et B : \n"); **scanf**("%d %d", &A, &B); **} while**(A>B); **printf**("Donner la valeur de N : \n"); **scanf**("%d", &N); S=0; P=1; **for** (i=1;i<=N;i++) **{** **do{** **scanf**("%d", &Vi); **} while** (Vi<A || Vi>B); **if** (Vi % 2==0) S=S + Vi; **else** P=P * Vi; **}** // Sortie **printf**("S = %d et P= %d", S,P) ; **}**


main.c [TP5] - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
<global>
main() : int
Management
Projects
Files
Workspace
TP5
Sources
main.c
main.c
#include<stdio.h>
int main() {
int A, B, N, i, Vi, S, P ;
// Entrées
do{
printf("Donner la valeur de A et B : \n");
scanf("%d %d", &A, &B);
} while(A>B);
printf("Donner la valeur de N : \n");
scanf("%d", &N);
S=0;
P=1;
for (i=1;i<=N;i++) {
do{
scanf("%d", &Vi);
} while (Vi<A || Vi>B);
if (Vi % 2==0)
S=S + Vi;
else
P=P * Vi;
}
// Sortie
printf("S = %d et P= %d", S,P) ;
}
C:\Users\Ahmed\Desktop\TP!
4 12
Donner la valeur de N :
5
6 10 5 7 5
S = 16 et P= 175
Process returned 0 (0x0) execution time : 18.383 s
Press any key to continue.
Après l'exécution