Solution

Exercise 1

```
#include <stdio.h>
```

Part a:

```
main ()
{
	int i, n, somme;

	somme = 0;

	i = 0;

	while(i<4)
	{
		printf("Saisir un entier : ");
		scanf("%d", &n);
		somme += n;

		i++;
	}

	printf ("La somme des entiers donnes ci - haut est : %d\n", somme);
}
```

En général pour calculer une somme qui s'incrémente peu à peu on l'initialise par son élément neutre de la somme qui est 0.

Initialisation de variable

Condition de la boucle

Incrément du variable (même que i=i+1)

Part b:

```
main ()
{
	int i, n, somme;
	somme = 0;
	i = 0;
	do
	{
		printf ("Saisir un entier : ");
		scanf("%d", &n);
		somme += n;
		i++;
	} while (i < 4);
	printf("La somme des entiers donnes ci - haut est : %d\n", somme);

}
```

(Presque même mais il' y a exécution de code Avant tester la condition) Cela est utilisé lors il y a condition sur la variable scanne ou saisie

Exemple d'exécution :

```
Saisir un entier : 5
Saisir un entier : 5
Saisir un entier : 4
Saisir un entier : 6
La somme des entiers donnes ci - haut est : 20
```

Exercise 2

```c
#include <stdio.h>

void main()
{
	int n, i;
	float result = 0;

	printf("donner n pour afficher ta suite! ");
	scanf("%d", &n);

	printf("la somme :");

	for (i = 1; i <= n; i++)
	{
		result += 1.0 / i;
		printf(" 1/%d +", i);
	}

	printf("est %f ", result);

}
```

- Result+= 1.0/i mettre 1.0 et non 1 est nécessaire car i est un entier et « 1 » un entier alors 1/i nous donne un entier même si la valeur est affectée dans un float result.
- Affichage dans chaque cycle de la boucle de 1/i.

Exemple d'exécution :

```
donner n pour afficher ta suite! 5
la somme : 1/1 + 1/2 + 1/3 + 1/4 + 1/5 +est 2.283334
```

Seulement pour un bon affichage on peut coder :

```c
void main()
{
	int n, i;
	float result = 0;
	printf("donner n pour afficher ta suite! ");
	scanf("%d", &n);
	printf("la somme :");

	for ( i = 1; i < n; i++)
	{
		result += 1.0 / i;
		printf(" 1/%d +", i);
	}

	printf("1/%d est %f ",i, result + 1.0/i);

}
```

Prenant le dernier élément a part et non afficher après lui un '+'.

Exemple d'exécution :

```
donner n pour afficher ta suite! 5
la somme : 1/1 + 1/2 + 1/3 + 1/4 +1/5 est 2.283333
```

```
donner n pour afficher ta suite! 10
la somme : 1/1 + 1/2 + 1/3 + 1/4 + 1/5 + 1/6 + 1/7 + 1/8 + 1/9 +1/10 est 2.928969
```

Exercice 3

```
#include <stdio.h>
void main()
{
	int nb=0,countmin=1,countmax=1;
	float sn = 0, n, max, min, moyenne=0;


	printf("Saisir une note (-1 pour terminer) : ");
	scanf("%f", &n);


	max = min = n;
	sn += n; nb++;

	while (n != -1)
	{
		printf("Saisir une note (-1 pour terminer) : ");
		scanf("%f", &n);


		if (n==-1)
			break;


		if (n > max)
		{
			max = n;
			countmax = 1;
		}
		else
			if (n == max)
				countmax++;

		if (n < min)
		{
			min = n;
			countmin = 1;
		}
		else
			if (n == min)
				countmin++;


		sn += n;
		nb++;
	}

	moyenne = sn / nb;

	printf("La moyenne des notes est %0.2f \nLa plus grande note est : %0.2f, cette
	note apparait %d fois dans la liste des notes \nLa plus petite note est : %0.2f,
	cette note apparait %d fois dans la liste des notes", moyenne, max, countmax,
	min, countmin);

}
```

Pour calculer le moyenne d'un nombre de notes inconnues on doit prendre en considération 2 variables :

- ✓ Une pour la somme de notes « sn »,
- ✓ Autre pour le nombre de note « nb » ;
- ✓ Puisqu'après on peut calculer le « moyenne = sn/nb. »

Initialiser « max » et « min » par la première note saisie pour faire de comparaison valide après avec les autres notes, ajouter « n » à « sn » et incrémenter « nb ».

Astucieux de mettre pour éviter des opérations sur une note invalide « -1 » qui peut causer une mal calcul de min et sn et nb.

Calcul de max :

- ❖ Si elle répète on ajoute un a la « counter »
- ❖ Si on trouve un nouveau max on reset « counter » a un et on affecte max pour la nouvelle valeur

Calcul de min :

- ❖ Semblable algorithme du max

Ajouter chaque « n » à « sn » et incrémenter

Exemple d'exécution :

```
Saisir une note (-1 pour terminer) : 12
Saisir une note (-1 pour terminer) : 15.25
Saisir une note (-1 pour terminer) : 13.5
Saisir une note (-1 pour terminer) : 7
Saisir une note (-1 pour terminer) : 11
Saisir une note (-1 pour terminer) : 12.5
Saisir une note (-1 pour terminer) : 7
Saisir une note (-1 pour terminer) : 9.75
Saisir une note (-1 pour terminer) : -1
La moyenne des notes est 10.86
La plus grande note est : 15.25, cette note apparait 1 fois dans la liste des notes
La plus petite note est : 7.00, cette note apparait 2 fois dans la liste des notes
```

Exercice 4

```c
#include <stdio.h>

void main()
{
	int n, i, test=1;

	printf("saisir un entier: ");
	scanf("%d", &n);

	for ( i = 2; i < n/2; i++)
	{
		if (n%i == 0)
		{
			test = 0;
			break;
		}
	}

	if (test==1)
	{
		printf("Cet entier est premier");
	}
	else
	{
		printf("Cet entier n'est pas premier");
	}

}
```

Test = 1. C.à.d. tout nombre est premier par défaut.

Si existe un nombre plus petit que n/2 qui divise n alors n n'est pas premier et soit dans ce cas test =0.

Affichage selon le cas.

Exemple d'exécution :

```
saisir un entier: 493
Cet entier n'est pas premier
```

```
saisir un entier: 379
Cet entier est premier
```

Exercise 5

```c
#include <stdio.h>
void main()
{
	int n, i;
	double Un, Un_1 = 1, Un_2 = 1;

	printf("saisir un entier: ");
	scanf_s("%d", &n);

	for (i = 3; i <= n; i++)
	{
		Un = Un_1 + Un_2;
		Un_2 = Un_1;
		Un_1 = Un;
	}


	if (n == 1 || n == 2)
		printf("L'entier numero %d de la suite de Fibonacci est %d", n, 1);

	else
		printf("L'entier numero %d de la suite de Fibonacci est %0.0lf", n, Un);
}
```

Après une calcule de « Un » c'est à savoir que prendre les variables dans une nouvelle exécution :
À noter que la valeur de « Un_2 » n'est pas plus utile d'où on l'affecte tout de suite sans la conserver par « Un_1 » puis de même « Un_1 » par « Un » et puis dans la nouvelle exécution « Un » désigne U_{n+1} d'où continuer suivant ce marche n-2 fois on obtient Un demandé au cas où n > 2. ***(*Voir note⬇)***

Exemple d'exécution :

```
saisir un entier: 9
L'entier numero 9 de la suite de Fibonacci est 34
```

```
saisir un entier: 50
L'entier numero 50 de la suite de Fibonacci est 12586269025
```

➡ ***Note :***

- L'ordre de l'affectation dans le boucle est très importante c'est désavantageux de commencer par « Un_1 =U_n » puis « Un_2 =Un_1 » car lors de la première affectation on a perdu la valeur de « Un_1 » qui est utile pour être « Un_2 ».
- D'où on commence l'affectation par ordre croissant des éléments de suites puisque dans un nouvelle exécution ce dernier élément (par exemple ici « Un_2 ») n'est pas plus utile.
- Exemple : pour une suite $U_n = 2 * U_{n-2} + 3 * U_{n-3}$; et $U_1 = U_2 = U_3 = 1$;
  - Il est rendre utile de déclarer une nombre de variables de U_{..} consécutif même si on ne l'utilise pas dans l'exécution de calcule mais utile pour la conservation de valeur.

- Allusion de solution :

```c
…
for (i = 4; i <= n; i++)
{
	Un = 2 * Un_2 + 3 * Un_3;
	Un_3 = Un_2;
	Un_2 = Un_1;
	Un_1 = Un;
}
…
```