UEP23 : Programmation renforcée
Licence STS - Deuxième année

Travaux Dirigés

Feuille 3

Exercice 1 et 2 :

- Faire un programme qui permette à l'utilisateur de remplir un tableau 1D de de taille inférieure à 15 contenant des entiers puis recherche si une valeur (rentrée par l'utilisateur) fait bien partie du tableau.
- Faire un programme qui permette à l'utilisateur de remplir deux tableaux 1D de taille variable (inférieure à 15) contenant des entiers. Le programme affichera l'intersection de ces deux tableaux.

// On suppose dispose aussi des fonctions des exo 5,6 et 7 du TD précédent

```
#include <stdio.h>

// Cette fonction recherche la valeur val dans le tableau tab.
// La taille max de tab est n
// Elle renvoie 1 si on a trouve, et 0 sinon
int isInTab(int *tab, int n, int val)  {
      int i;
      for (i=0; i<n;i++) {
            if (tab[i] == val)
                  return 1;      // si on trouve, on s'arrete.
      }
      // Si on est la, c'est qu'on a pas trouve
      // (sinon, on serait sorti avant !)
      // On peut donc renvoyer 0
      return 0;
}
```

```
void intersectionTableaux(int *tab1, int n1, int *tab2, int n2, int *tabInter, int *nInter) {
	int i;  // pour se deplacer dans le tableau 1;

	*nInter = 0;    // A debut, le tableau d'intersection est vide !
	for (i=0;i<n1;i++) {
		// On verifie si le contenu de la case i du tableau 1
		// existe dans le tableau 2
		if (isInTab(tab2,n2,tab1[i]) == 1)  {
			// Il faut l'ajouter au tableau d'intersection
			// SAUF si ellee y est deja !
			if (isInTab(tabInter,*nInter,tab1[i]) == 0 ) {
				// On l'ajoute au tableau
				tabInter[*nInter] = tab1[i];
				// Qui contient maintenant une case de plus !
				(*nInter)++;
			}
		}
	}
}

int main (void)
{
	int tab1[15], tab2[15],tab3[15];
	int n1,n2,n3;

	printf("Tableau 1 : Sur combien de cases voulez vous travailler ?\n");
	scanf("%d",&n1);
	remplirTableau(tab1,n1);
	afficherTableau(tab1, n1);

	printf("Tableau 2 : Sur combien de cases voulez vous travailler ?\n");
	scanf("%d",&n2);
	remplirTableau(tab2,n2);
	afficherTableau(tab2, n2);

	intersectionTableaux(tab1,n1,tab2,n2,tab3, &n3);
	printf("l'intersection stricte de ces tableaux est :\n");
	afficherTableau(tab3, n3);

	return 1;
}
```

Exercice 2 :

Exercice 3 :

Sans utiliser de tableaux, faire un programme qui lise un entier n au clavier puis affiche des « X » en triangle sur n lignes. Par exemple, si l'utilisateur a rentré le chiffre 3, le programme affichera :

X
XX
XXX

Réaliser la même chose affichant un damier de « X »

```
#include <stdio.h>

int main(void)
{
	int i;  // Pour se deplacer sur les differentes lignes;
	int j;  // Pour se deplacer sur les differentes colonnes d'une ligne;
	int n;  // le nombre de lignes;

	printf("Combien de lignes voulez vous ? \n");
	scanf ("%d",&n);

	printf ("Le triangle de X : \n");
	for (i=1;i<=n;i++) {
		// sur la ligne i, on affiche i "X"
		for (j=1;j<=i;j++)
			printf("X");
		// Quand on a fini avec les X on passe a la ligne
		printf("\n");
	}

	printf ("\n\nLe damier de X : \n");
	for (i=1;i<=n;i++) {
		// On peut le faire avec des if (ligne paire, colonne impaire....)
		// ou remarquer qu'il faut un X si
		// (i+j)%2 ==0
		for (j=1;j<=n;j++)
			if ( (i+j)%2 ==0 )
				printf("X");
			else
				printf(" ");

		// Quand on a fini avec les X on passe a la ligne
		printf("\n");
	}

	return 1;
}
```

Exercice 4 :

Faire une fonction permettant de remplir un triangle de pascal de N lignes (N inférieur à 15).

```
#include <stdio.h>

int main(void)
{
	int i;  // Pour se deplacer sur les differentes lignes;
	int j;  // Pour se deplacer sur les differentes colonnes d'une ligne;
	int n;  // le nombre de lignes;
	int tab[50][50]; // On prend un gros tableau !

	printf("Combien de lignes voulez vous ? \n");
	scanf ("%d",&n);

	// On met des zeros partout
	for (i=0;i<n;i++)
		for (j=0;j<n;j++)
			tab[i][j]=0;

	// On remplit la premiere colonne de 0
	for (i=1;i<=n;i++)
		tab[i][1]=1;

	// On remplit la diagonale de 1 (C(i,i) =1;
	for (i=1;i<=n;i++)
		tab[i][i]=1;

	// On utilise sur les cases restantes la relation
	// de recurrence C(i,j)=c(i-1,j-1) + C(i-1,j)
	for (i=2;i<=n;i++)
		for (j=2;j<i;j++)
			tab[i][j] = tab[i-1][j-1]+tab[i-1][j];

	// On affiche le tout
	printf ("Le triangle de Pascal : \n");
	for (i=1;i<=n;i++) {
		for (j=1;j<=i;j++)
			printf("%d ",tab[i][j]);
		printf("\n");
	}

	return 1;
}
```