DS n° 3 - Informatique I

Calculatrice et documents non autorisés

Lorsqu'aucune consigne n'est précisée, les réponses peuvent être données en langage C ou en pseudo-code. Prêtez une attention particulière à la robustesse de vos programmes.

Exercice 1 (*Convertisseur binaire / décimal*) **(3.00 pts)**
(Cet exercice doit être codé en langage C obligatoirement).

1. Écrire une procédure `void afficheBinaire(unsigned int num)` qui prend en paramètre un nombre entier positif ou nul, et qui affiche la valeur correspondante au format binaire (notez qu'on ne demande pas de stocker ce nombre binaire dans une variable, simplement de l'afficher).
2. Écrire une fonction `unsigned int binaireToDecimal(char bin[])` qui prend en paramètre une chaîne de caractères dont chaque caractère représente le chiffre d'un nombre binaire (par exemple : `bin = "101101"`). Cette fonction doit convertir cette chaîne de caractères en entier et retourner la valeur convertie.
   Cette fonction doit stopper le programme avec un code d'erreur si elle rencontre le moindre problème.

Exercice 2 (*Encodeur Binaire 4 vers 2*) **(4.25 pts)**
Concevoir un encodeur binaire qui prend quatre entrées binaires (A3, A2, A1, A0) et produit deux sorties (S1, S0) en indiquant la position du bit le plus significatif (bit de poids fort) à 1.
Ici A3 est le bit de poids fort et A0 le bit de poids faible.
Idem S1 est le bit de poids fort et S0 le bit de poids faible.

Les sorties (S1, S0) indiqueront la position du bit le plus significatif à 1 en binaire. (de la position 0 à la position 3, respectivement de la droite à la gauche).
Les sorties S1 et S0 doivent être codées en binaire naturel (00 pour 0, 01 pour 1, 10 pour 2, 11 pour 3).
Exemples :
— Entrée : 0101
→ S1 = 1, S0 = 0 → le bit de poids le plus fort avec une valeur à 1 se trouve à l'index '2' (la $3^{\text{ème}}$ place).
— Entrée : 0011
→ S1 = 0, S0 = 1 → le bit de poids le plus fort avec une valeur à 1 se trouve à l'index '1' (la $2^{\text{ème}}$ place).
Respecter les étapes suivantes :

1. Écrire la table de vérité du convertisseur binaire.
2. Réaliser le tableau de Karnaugh associé.
3. On ne sait pas quelle valeur de sortie S mettre pour une entrée A qui vaut 0 : donner la valeur de S1 et S0 pour A=0, permettant ainsi de simplifier les equation de sortie le plus possible.
4. En déduire les équations réduites.
5. Dessiner le circuit logique associé aux équations.

Exercice 3 (*Manipulation de Polynômes*) **(5.25 pts)**
(Cet exercice doit être codé en langage C obligatoirement).
Dans cet exercice, nous manipulons des polynômes dont les coefficients sont représentés dans un tableau. L'indice du tableau correspond au degré associé à chaque coefficient. Par exemple, le polynôme $8x^4 - 9x^2 + 7,6x + 5$ est représenté par le tableau $[5, 7.6, -9, 0, 8]$ (où la valeur 0 indique l'absence de coefficient pour le degré 3).

1. Écrire une fonction ou procédure `displayPoly(...)` qui, prenant en paramètres un tableau et sa taille, affiche le polynôme correspondant.
   Exemple : [5, 7.6, −9, 0, 8] sera affiché

   ```
   +8x^4 -9x^2 +7.6x + 5
   ```

   .
2. Écrire une fonction ou procédure `evaluationPoly(...)` qui prend en paramètre :

— Un tableau,
— Sa taille,
— Un réel représentant la valeur de x,
et qui retourne le résultat du polynôme pour cette valeur de x.
(On suppose l'existence de la fonction `float pow(float x, float y);` en langage C, qui permet de calculer x^y).

3. Écrire une fonction ou procédure `addPoly(...)` pour additionner deux polynômes de mêmes degrés. Ce bout de programme prendra en paramètre deux tableaux de mêmes tailles et mettra dans le premier la somme des deux polynômes.
4. Écrire une fonction ou procédure `derivativePoly(...)` permettant de dériver un polynôme passé en paramètre. Ce bout de programme prendra en paramètre 1 tableau et sa taille et le modifiera en suivant le calcul de la dérivée d'un polynome.

Exercice 4 (*Remplissage d'un tableau 2D*) **(3.50 pts)**
On souhaite remplir un tableau à 2 dimensions de taille de côté N avec des nombres entiers croissants de 1 à N^2. En partant d'un tableau initialisé avec des valeurs à 0, le remplissage se fera en forme de spirale avec la méthode suivante :

1. démarrage par la case en haut à gauche avec la valeur 1
2. on se déplace vers la droite jusqu'à rencontrer le bord droit de la matrice ou bien une valeur différente de 0
3. on se déplace vers le bas jusqu'à rencontrer le bord inférieur de la matrice ou bien une valeur différente de 0
4. on se déplace vers la gauche jusqu'à rencontrer le bord gauche de la matrice ou bien une valeur différente de 0
5. on se deplace vers le haut jusqu'à rencontrer le bord supérieur de la matrice ou bien une valeur différente de 0
6. incrémenter la valeur de la case de +1 par rapport à la précédente case, et continuer ces etapes en reprenant à l'etape 2 jusqu'à atteindre la valeur N^2 (ce qui devrait concorder avec le fait que toutes les cases du tableau ont été remplies).

Pour un tableau de taille de côté $N = 5$, on a le schéma ci-dessous qui indique dans quel état doit se trouver le tableau après exécution du programme.

1	2	3	4	5
16	17	18	19	6
15	24	25	20	7
14	23	22	21	8
13	12	11	10	9

Écrire un programme principal qui va déclarer un tableau à deux dimensions de taille carrée TAILLE définie comme constante.
Le programme doit initialiser le tableau avec des 0, et ensuite le remplir comme indiqué précédemment.
Ce bout de programme ne doit rien faire d'autre que le remplissage du tableau.

Exercice 5 (*Vérification de Mot de Passe*) **(4.00 pts)**

Écrire une fonction ou procédure `checkPassword(...)` qui prend en entrée une chaîne de caractères qui est un mot de passe entré par un utilisateur.
Cette fonction retourne en sortie une valeur entière qui indique si le mot de passe respecte les règles de sécurité. Pour vous aider, la table ASCII est fournie en annexe.
Les règles de sécurité pour le mot de passe sont les suivantes :
— La longueur du mot de passe doit être d'au moins 8 caractères,
— Le mot de passe doit contenir au moins 1 lettre minuscule,
— Le mot de passe doit contenir au moins 1 lettre majuscule,
— Le mot de passe doit contenir au moins 1 chiffre,
— Le mot de passe doit contenir au moins 1 caractère spécial dont la valeur dans la table ASCII est strictement supérieure à 20 (base 16) et strictement inférieur à 7F (base 16).
La fonction doit retourner une valeur entière dont la valeur dépend du contenu du mot de passe donné en entrée :
— 0 : le mot de passe respecte toutes les règles de sécurité,
— 1 : le mot de passe a une mauvaise longueur,
— 2 : le mot de passe a une longueur correcte mais l'un de ses caractères n'est pas autorisé,
— 3 : le mot de passe a une longueur correcte et tous ses caractères sont autorisés mais il ne respecte pas l'ensemble des règles de sécurité énoncées plus haut.
Il est interdit d'utiliser les fonctions du module <string.h> dans cet exercice.

[ANNEXE : table ASCII]

Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-_	63	3F	?	95	5F	_	127	7F	DEL