Commencé le	mardi 10 novembre 2020, 09:00
État	Terminé
Terminé le	mardi 10 novembre 2020, 09:54
Temps mis	54 min 22 s
Note	**18,50** sur 20,00 (**93**%)

Question **1**

Correct

Note de 4,00 sur 4,00

Écrire la fonction **int bitcount(unsigned int nb)** qui affiche le nombre de bits à 1 dans la représentation binaire de l'entier **nb.**

Par exemple,

- bitcount(10) renvoie 2 (car 10 est représenté par 1010 en binaire)
- bitcount(4) renvoie 1 (car 4 est représenté par 100 en binaire)

Par exemple:

Test	Résultat
`printf("bitcount(10) = %d\n", bitcount(10));` `printf("bitcount(2) = %d\n",  bitcount(2));`	`bitcount(10) = 2` `bitcount(2) = 1`
`for (int i = 10; i < 10; i++)` `    printf("bitcount(%d) = %d\n", i, bitcount(i));`	

Réponse : (régime de pénalités : 10, 20, ... %)

Réinitialiser la réponse

```
int bitcount(unsigned int n)
{
    int res = 0;
    while (n)
    {
        res += 1;
        n &= n - 1;
    }
    return res;
}

```

	Test	Résultat espéré	Got	
✔	`printf("bitcount(10) = %d\n", bitcount(10));` `printf("bitcount(2) = %d\n",  bitcount(2));`	`bitcount(10) = 2` `bitcount(2) = 1`	`bitcount(10) = 2` `bitcount(2) = 1`	✔
✔	`for (int i = 10; i < 10; i++)` `    printf("bitcount(%d) = %d\n", i, bitcount(i));`			✔

Tous les tests ont été réussis ! ✔

Correct

Note pour cet envoi : 4,00/4,00.

Question **2**

Correct

Note de 4,00 sur 4,00

Écrire un programme C qui supprime les caractères compris entre les colonnes **MIN** et **MAX** de chaque ligne du fichier standard d'entrée. Le résultat est affiché sur le fichier standard de sortie.

Le premier caractère est, par convention ici, situé à l'indice 1.

Dans les exemples qui suivent, **MIN** et **MAX** ont été fixés à 5 et 9.

Par exemple:

Entrée	Résultat

`0 1 2 3`
`12345678901234567890`
`abcde`
	`0 1 2 3`
	`12340123456789012345678 90`
	`abcd`

`Une ligne de texte` `et une autre ligne ....`	`Une de texte` `et utre ligne ....`

`1 ligne`
`et`
`1 ligne un peu plus longue`
`se`
`suivent`
`.....`
	`1 li`
	`et`
	`1 lin peu plus longue`
	`se`
	`suiv`
	`.....`

Réponse : (régime de pénalités : 10, 20, ... %)

Réinitialiser la réponse

```c
#include <stdio.h>

#define MIN 5
#define MAX 9

int main()
{
    int col = 0;
    char c;
    while ((c = getc(stdin)) != EOF)
    {
        col++;
        if (c == '\n')
            col = 0;
        else if (col >= MIN && col <= MAX)
            continue;
        putchar(c);
    }
    return 0;
}
```

	Entrée	Résultat espéré

✔	`0 1 2 3`
	`12345678901234567890`
	`abcde`
		`0 1 2 3`
		`12340123456789012345678 90`
		`abcd`

✔	`Une ligne de texte` `et une autre ligne ....`	`Une de texte` `et utre ligne ....`

✔	`1 ligne`
	`et`
	`1 ligne un peu plus longue`
	`se`
	`suivent`
	`.....`
		`1 li`
		`et`
		`1 lin peu plus longue`
		`se`
		`suiv`
		`.....`

Tous les tests ont été réussis ! ✔

Correct

Note pour cet envoi : 4,00/4,00.

Question **3**

Terminer

Note de 3,00 sur 4,00

Écrire la fonction **suppression(str, i, j)** qui supprime les caractères de la chaîne **str** entre les indices **i** et **j** (compris).

Eventuellement, si les indices i et j sont en dehors de l'intervalle **[0 .. srtlen(str)[** , ils seront ramenés à la borne valide la plus proche.

Vous ne devez pas passer par un tableau intermédiaire.

Par exemple:

Test	Résultat

`char s[] = "0123456789";`
`suppression(s, 0, 5);`
`printf("%s\n", s);`
	`6789`

`char s[] = "0123456789";`
`suppression(s, 1, 8);`
`printf("%s\n", s);`
	`09`

`char s[] = "0123456789";`
`suppression(s, -3, 100);`
`printf("%s\n", s);`	

Réponse : (régime de pénalités : 10, 20, ... %)

Réinitialiser la réponse

```
void suppression(char str[], int i, int j)
{
    int len = strlen(str);
    if (j <= 0 || i > len)
        return;
    if (len < ++j)
        j = len;
    if (i < 0)
        i = 0;
    if (i > j)
        return;
    while (j <= len && (str[i++] = str[j++]));
}
```

Partiellement correct

Points pour cet envoi : 3,20/4,00. En tenant compte des tentatives précédentes, cela donne **2,56/4,00**.

Commentaire :

Question **4**

Correct

Note de 4,00 sur 4,00

Écrire la fonction **int nombre_Z(int nb)** qui renvoie 1 si le nombre **nb** est divisible par le produit de ses chiffres, et 0 sinon.

Cette fonction devra renvoyer 1 pour les nombres suivants:

- **15** car divisible par 5 (1x5),
- **24** car divisible par 8 (2x4**)** ou
- **312** car divisible par 6 (3x1x2).

A l'inverse, elle renverra 0 pour les nombres

- **25** (pas divisible par 10) ou
- **99** (pas divisible par 81)

Par exemple:

Test	Résultat
`printf("nombre_Z(216) = %d\n", nombre_Z(216));` `printf("nombre_Z(89) = %d\n", nombre_Z(89));`	`nombre_Z(216) = 1` `nombre_Z(89) = 0`
`for (int i = 10; i < 100; i++)` `  if (nombre_Z(i)) printf("%d\n", i);`	11 12 15 24 36

Réponse : (régime de pénalités : 10, 20, ... %)

Réinitialiser la réponse

```
int nombre_Z(int n)
{
    if (n == 0)
        return 0;
    if (n < 0)
        n = -n;
    int prod = 1;
    for (int k = n; k; k /= 10)
    {
        prod *= k % 10;
    }
    return prod != 0 && n % prod == 0;
}
```

	Test	Résultat espéré	Got	
✔	`printf("nombre_Z(216) = %d\n", nombre_Z(216));` `printf("nombre_Z(89) = %d\n", nombre_Z(89));`	`nombre_Z(216) = 1` `nombre_Z(89) = 0`	`nombre_Z(216) = 1` `nombre_Z(89) = 0`	✔
✔	`for (int i = 10; i < 100; i++)` `  if (nombre_Z(i)) printf("%d\n", i);`	11 12 15 24 36	11 12 15 24 36	✔

Tous les tests ont été réussis ! ✔

Correct

Note pour cet envoi : 4,00/4,00.

Question **5**

Terminer

Note de 3,50 sur 4,00

Écrire la fonction **imprimer** qui permet d'imprimer une série de paramètres de longueur impaire. Les paramètres d'indice impair sont des chaînes de caractères et les paramètres d'indice pair sont de type entier. La série de paramètres se termine par la valeur **NULL**.

Cette fonction permet d'imprimer une série d'entiers où chaque entier est précédé par une chaîne le décrivant. Ainsi par exemple l'appel

```
imprimer("Valeur de x=", x, " Valeur de y=", y, " => x+y=", x+y, NULL);
```

affichera sur la sortie standard.

```
Valeur de x=1 Valeur de y=2 => x+y=3
```

si **x** et **y** valent **1** et **2.**
Par exemple:

Test	Résultat
`int x=1, y = 2;` `imprimer("Valeur de x=", x, " Valeur de y=", y, " => x+y=", x+y, NULL);`	`Valeur de x=1 Valeur de y=2 => x+y=3`
`int x=1, y = 2;` `imprimer("Valeur de x+y=", x+y, NULL);`	`Valeur de x+y=3`

Réponse : (régime de pénalités : 10, 20, ... %)

Réinitialiser la réponse

```c
#include <stdarg.h>

void imprimer(char *str, ...)
{
    va_list args;
    va_start(args, str);

    char* ptr = str;
    while (ptr)
    {
        printf("%s%d", ptr, va_arg(args, int));
        ptr = va_arg(args, char*);
    }

    va_end(args);
}
```

	Test	Résultat espéré	Got	
✔	`int x=1, y = 2;` `imprimer("Valeur de x=", x, " Valeur de y=", y, " => x+y=", x+y, NULL);`	`Valeur de x=1 Valeur de y=2 => x+y=3`	`Valeur de x=1 Valeur de y=2 => x+y=3`	✔
✔	`int x=1, y = 2;` `imprimer("Valeur de x+y=", x+y, NULL);`	`Valeur de x+y=3`	`Valeur de x+y=3`	✔

Tous les tests ont été réussis ! ✔

Partiellement correct

Points pour cet envoi : 4,00/4,00. En tenant compte des tentatives précédentes, cela donne **3,20/4,00**.

Commentaire :

Aller à...

QCM 08/12 ►