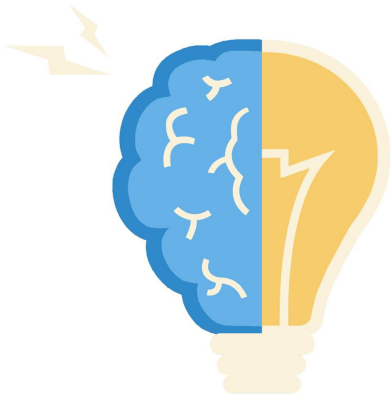
BRIGHT CODE

SUCCES MODUS OPERANDI

{ DAY 01 }

Responsable D01: Raphaël Moisset

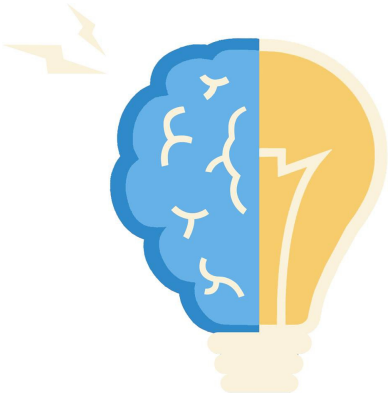


{ INSTRUCTIONS }

Rendu : http://github.com/Romanovytch/login.git

Compilation : gcc *.c -lbc -L.

Binaires de test : ---- à 23h ----



Les Variables (1/2)

Exercice 01 ~ mirror_tell_me()

Faire une fonction **mirror_tell_me** qui ajoute ou soustrait le bon nombre à chaque caractères du mot «brightcode» pour afficher «phenomenal». Puis, on affiche un saut à la ligne.

Notions

- Variables
- Fonctions
- Table ASCII

Fonctions autorisées de la librairie brightcode.h:

- bc_write_char();

/!\ L'utilisation d'une quelconque autre fonction est prohibée.

Prototype de la fonction : **void mirror_tell_me(void);**
Nom du dossier : **var_ex01**

Les Variables (2/2)

Exercice 02 ~ addition(), subtraction(), multiplication()

Faire les fonctions **addition, subtraction** et **multiplication** qui respectivement additionnent, soustraient et multiplient deux entiers passés en paramètres et retournent le résultat.

/!\ Le résultat des opérations effectuées par la contrôleuse ne dépasseront pas les bornes d'un int. On peut donc aisément stocker le résultat dans un int.

Notions

- Variables
- Fonctions

Fonctions autorisées de la librairie brightcode.h:

- Aucune

/!\ L'utilisation d'une quelconque autre fonction est prohibée.

Prototype des fonctions :
int addition(int, int);
int subtraction(int, int);
int multiplication(int, int);
Nom du dossier : **var_ex02**

Conditions (1/2)

Exercice 01 ~ age_if(), age_case()

Faire les fonctions **age_if()** et **age_case()** qui prennent en paramètre un entier signé qui représente l'âge d'un individu. On appliquera les conditions suivantes à cet âge:

S'il est égal à 0, on affiche «Good Joke !»
Sinon, s'il est égal à 5, on affiche «Too Young !»
Sinon, s'il est égal à 12, on affiche «Child !»
Sinon, s'il est égal à 16, on affiche «Teenager !»
Sinon, s'il est égal à 18, on affiche «You're an adult !»
Sinon, on affiche un retour à la ligne.

Le comportement des fonctions est identique, à l'exception que les conditions de age_if() sont rédigées à l'aide de «if, else if, else», tandis que les conditions de age_case() sont rédigées à l'aide de «switch, case, break, default»

Notions

- Variables
- Fonctions
- Conditions
- Opérateurs de comparaison

Fonctions autorisées de la librairie brightcode.h:

- bc_write_string();

/!\ L'utilisation d'une quelconque autre fonction est prohibée.

Prototype de la fonction :
void age_if(int);
void age_case(int);
Nom du dossier : **if_ex01**

Conditions (2/2)

Exercice 02 ~ bank()

Faire une fonction **bank()** qui prend en paramètre deux entiers signés qui représentent respectivement l'âge et l'argent d'un individu. On appliquera les conditions suivantes à ces variables :

Si l'âge est inférieur à 18 ou l'argent est inférieur à 1000
    On affiche «You can't enter the bank.»
Sinon, si l'âge est supérieur à 70 et l'argent est supérieur ou égal à 100 000
    On affiche «Make a placement...»
Sinon, si l'argent est supérieur ou égal à 100 000
    On affiche «WELCOME, HAVE A SIT !»
Sinon, si l'argent est supérieur ou égal à 10 000
    On affiche «Welcome.»
Sinon
    On affiche «You can enter the bank.»

Les phrases affichées se terminent par un retour à la ligne.

Notions

- Variables
- Fonctions
- Conditions
- Opérateurs de comparaison, opérateurs logiques

Fonctions autorisées de la librairie brightcode.h:

- bc_write_string();

/!\ L'utilisation d'une quelconque autre fonction est prohibée.

Prototype de la fonction : **void bank(int, int);**
Nom du dossier : **if_ex02**

Les Boucles (1/2)

Exercice 01 ~ disp_alpha()

Faire une fonction **disp_alpha()** qui affiche l'alphabet à l'endroit, suivit d'un retour à la ligne, puis l'alphabet à l'envers, suivit d'un retour à la ligne.

Notions

- Variables
- Fonctions
- Table ASCII
- Boucles

Fonctions autorisées de la librairie brightcode.h:

- bc_write_char();

/!\ L'utilisation d'une quelconque autre fonction est prohibée.

Prototype de la fonction : **void disp_alpha(void);**
Nom du dossier : **loop_ex01**

Défense : Vous serez également évalué sur la pertinance du type de boucle utilisé.

Les Boucles (2/2)

Exercice 02 ~ star_pyramid()

Faire une fonction **star_pyramid()** qui prend en paramètre un entier non signé qui représente la hauteur de la pyramide. La fonction affichera à l'écran la pyramide en utilisant le caractère '*'. Dans le cas de 0, on affichera un retour à la ligne.

Exemple :

- star_pyramid(5) affiche

```
qwarky@solaris:~/workspace/icp_day01/loop_ex02 $ ./a.out
    *
   ***
  *****
 *******
*********
```

Notions

- Variables
- Fonctions
- Table ASCII
- Boucles

Fonctions autorisées de la librairie brightcode.h:

- bc_write_char();

/!\ L'utilisation d'une quelconque autre fonction est prohibée.

Prototype de la fonction : **void star_pyramid(unsigned int);**
Nom du dossier : **loop_ex02**

Défense : Vous serez également évalué sur la pertinance du type de boucle utilisé.

BRIGHT CODE

SUCCES MODUS OPERANDI

{ ALGORITHMIE }

- D1 → ave_cesar()

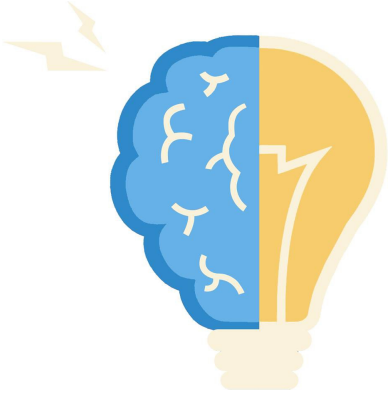
★

- D2 → reverse_number()

★ ★

D3 → dec_to_hex()

★★★



Algorithmie D1

Exercice 01 ~ ave_cesar()

Faire une fonction **ave_cesar()** qui prend en paramètre un entier non signé. La fonction affichera cet entier en chiffre romain, selon la convention suivante :

I : 1, V : 5, X : 10, L : 50, C : 100, D : 500, M : 1000

On emploiera l'ancienne notation romaine, c'est-à-dire, sans soustraction d'unités. Exemple :

ave_cesar(9) affiche :
VIIII

ave_cesar(900) affiche :
DCCCC

On affichera un retour à la ligne pour le cas du 0.

Notions

- Variables
- Fonctions
- Table ASCII
- Conditions
- Boucles

Fonctions autorisées de la librairie brightcode.h:

- bc_write_char();

/!\ L'utilisation d'une quelconque autre fonction est prohibée.

Prototype de la fonction : **void ave_cesar(unsigned int);**
Nom du dossier : **algo_ex01**

Algorithmie D2

Exercice 02 ~ reverse_number()

Faire une fonction **reverse_number()** qui prend en paramètre un entier signé. La fonction affichera cet entier à l'envers.

Notions

- Variables
- Fonctions
- Table ASCII
- Conditions
- Boucles
- Bases décimal et hexadécimal

Fonctions autorisées de la librairie brightcode.h:

- bc_write_char();

/!\ L'utilisation d'une quelconque autre fonction est prohibée.

Prototype de la fonction : **void reverse_number(int);**
Nom du dossier : **algo_ex02**

Algorithmie D3

Exercice 03 ~ dec_to_hex()

Faire une fonction **dec_to_hex()** qui prend en paramètre un entier non signé. La fonction affichera cet entier en hexadecimal, c'est-à-dire, en base 16.

Notions

- Variables
- Fonctions
- Table ASCII
- Conditions
- Boucles
- Bases décimal et hexadécimal

Fonctions autorisées de la librairie brightcode.h:

- bc_write_char();

/!\ L'utilisation d'une quelconque autre fonction est prohibée.

Prototype de la fonction : **void dec_to_hex(unsigned int);**
Nom du dossier : **algo_ex03**