PRÉNOM: Raquel

Programmation C

Révisions

Question 1:

f(1, 608) {

y=2 x 2; print +(1, 4);

d=2+4; print +(6, 5);

y=5;

3

x=3

Indiquez les valeurs imprimées sur la sortie standard.

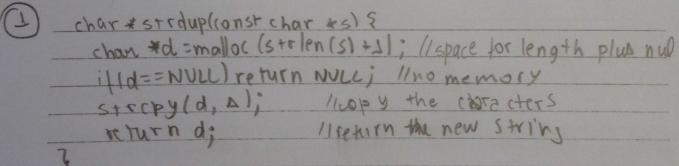
24 65 15

Question 2:

Ecrire la fonction

char *strdup(const char *s);

qui crée une copie de la chaîne de caractères s. Cette fonction renvoie un pointeur sur la copie si celle-ci a pu se faire et NULL dans le cas contraire.



(2) char *Str dup(cox+ char * S) {

if(Strlen(A) = =0) return NULL;

chan * Str_copy = malloc(signaf(a));

chan * ptn-temp_A = A;

chan * ptn-temp_ copy = htr_copy;

chan * ptn_temp_ copy = htr_copy;

while(* ptn_temp_a)*(ptr_temp_copy++) = * (ptn_temp_a++)

return Str_copy);

Question 3:

Écrire le programme cmp fich1 fich2

qui compare les fichiers fichier1 et fichier2. Si les deux fichiers sont identiques le programme n'affiche rien, sinon il affiche le caractère et la ligne où la différence a été trouvée avant de s'arrêter. Pour simplifier, on suppose ici que le programme est bien appelé avec deux fichiers et que ceux-ci existent et sont lisibles.

regivel int cmp (har * file I name, char * file 2-name) { FILE* file 1, *lik 2; file = fopen (files - name, "n"); sile 2 = fopen (file 2 - name, "r") while ((C) = fgetc/file 1) != EOF && (ca = fgetc/file2) != EOF) if (Es! = car return 0; 3 int main (intunge, their const * argy[]) ? if (storcmplangv[1], "cmp")=0,9

check-argumentlangc, argv);

print; ("is diff? ".d in", cmplangv[2], arg v[3]); puts (read-line (ong v)); return 0;

Question 4:

Ajouter les contrôles nécessaires à une bonne exécution du programme cmp précédent. On ne veut ici que la partie qui contrôle les paramètres. Inutile de recopier ici votre réponse à la question précédente.

int check-argument (intronge, consistent ** pString) {

if (ange < 9 & parge > 4) return 0,

if (stremple string[s], "emp)! = 0) return 0;

FILE ** Like J, ** | like 2,

lik J = lopen (pString[2], "r");

fik 2 : fopen (pString[3], "r");

if ! ! like J | ! ! like 2) return 0;

return J;

Question 5:

Écrire le programme

head [-n] [fichier]

qui copie les n premières lignes de fichier sur la sortie standard. Si fichier n'est pas spécifié, head prend ses lignes sur l'entrée standard. La valeur par défaut pour n est de 10 lignes.

On rappelle que la fonction int atoi(const char *str); peut être utilisée pour convertir la chaîne de caractères str en entier.

define BUFFER _MAX-LENGTH 1029

char* read_line(char comm * argy[]){

int nline = a toi(argy[3]);

File * lile = fopen(argy[4], "n");

j(! file) return NULL;

char copy[BUFFER_MAX-LENGTH];

char line [BUFFER_MAX-LENGTH];

while I gets (line, size of (line), file) && nline >0) &

Str(at (lopy, line);

Pline --;

return uppy;