Algoritmos y Programación I (95.11) – Curso Kuhn – 2do parcialito, 2do recuperatorio – 04/07/2019

Resolver los siguientes problemas en forma clara y legible en código ISO-C99.

1. Una persona se representa según un *nombre* que es una cadena de no más de `MAX_CADENA` elementos, un *DNI* que es un entero sin signo, un *año de nacimiento* que es un entero corto sin signo y un *padre* y una *madre* que son dos referencias a personas.

   a. Declarar una estructura `struct persona` que modele a la persona descripta.

   b. Definir el tipo `persona_t` en base a la estructura anterior.

   c. Escribir una función `bool tiene_padres_mayores(const persona_t *p, short edad);` que indique si alguno de los padres era mayor de `edad` años al momento de nacer la persona `p`. Si no hubiera referencia se asumirá que no era mayor a dicha edad.

2. a. Implementar una función

   ```
   char **clonar_arreglo_cadenas(char *ss[]);
   ```

   que reciba un arreglo de cadenas de caracteres `ss` el cual contiene cadenas en sus primeras posiciones y el centinela `NULL` en su última posición y que devuelva una copia en memoria nueva de dicho arreglo.

   b. Implementar una función

   ```
   void destruir_arreglo_cadenas(char *ss[]);
   ```

   que reciba un arreglo dinámico de cadenas dinámicas como el del ítem anterior y libere la memoria asociada a él.

3. Escribir una función `bool leer_flotantes(float **pv, size_t *n);` que lea valores flotantes de `stdin` y los almacene en un vector dinámico de flotantes. Los parámetros `pv` y `n` son sólamente de salida y la función debe retornar a través de ellos el vector dinámico y su longitud. A su vez debe retornar por el nombre `true` si todo está bien o `false` en caso de error.

¡Suerte! :)

Algoritmos y Programación I (95.11) – Curso Kuhn – 2do parcialito, 2do recuperatorio – 04/07/2019

Resolver los siguientes problemas en forma clara y legible en código ISO-C99.

1. Una persona se representa según un *nombre* que es una cadena de no más de `MAX_CADENA` elementos, un *DNI* que es un entero sin signo, un *año de nacimiento* que es un entero corto sin signo y un *padre* y una *madre* que son dos referencias a personas.

   a. Declarar una estructura `struct persona` que modele a la persona descripta.

   b. Definir el tipo `persona_t` en base a la estructura anterior.

   c. Escribir una función `bool tiene_padres_mayores(const persona_t *p, short edad);` que indique si alguno de los padres era mayor de `edad` años al momento de nacer la persona `p`. Si no hubiera referencia se asumirá que no era mayor a dicha edad.

2. a. Implementar una función

   ```
   char **clonar_arreglo_cadenas(char *ss[]);
   ```

   que reciba un arreglo de cadenas de caracteres `ss` el cual contiene cadenas en sus primeras posiciones y el centinela `NULL` en su última posición y que devuelva una copia en memoria nueva de dicho arreglo.

   b. Implementar una función

   ```
   void destruir_arreglo_cadenas(char *ss[]);
   ```

   que reciba un arreglo dinámico de cadenas dinámicas como el del ítem anterior y libere la memoria asociada a él.

3. Escribir una función `bool leer_flotantes(float **pv, size_t *n);` que lea valores flotantes de `stdin` y los almacene en un vector dinámico de flotantes. Los parámetros `pv` y `n` son sólamente de salida y la función debe retornar a través de ellos el vector dinámico y su longitud. A su vez debe retornar por el nombre `true` si todo está bien o `false` en caso de error.

¡Suerte! :)