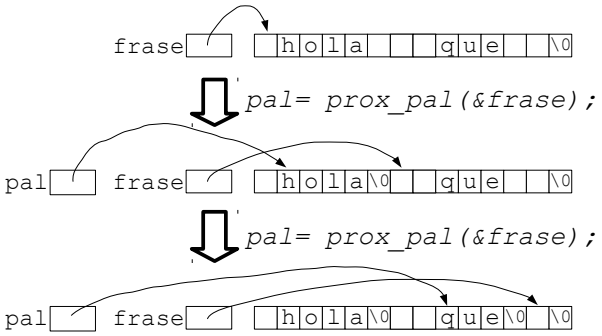
CC3301 Programación de Software de Sistemas
Examen – Semestre Primavera 2016 – Prof.: Luis Mateu

Pregunta 1 (40%)

Parte a.- Programe la siguiente función:

```
char *prox_pal(char **pfrase);
```

El parámetro **pfrase* apunta a un string de C que contiene múltiples palabras separadas por uno o más espacios en blanco. Esta función debe retornar la primera palabra de la frase y entregar en **pfrase* lo que quedó de la frase. La siguiente figura muestra 2 ejemplos de uso. Los punteros *pal* y *frase* son de tipo *char**. Inicialmente *frase* apunta al string `" hola que "` y termina apuntando a `" "`.



La primera llamada retorna `"hola"` y la segunda `"que"`. Observe que Ud. debe modificar el string recibido colocando la terminación de string (`'\0'`) para la palabra retornada. Si no quedan palabras en *frase*, se debe retornar *NULL*.

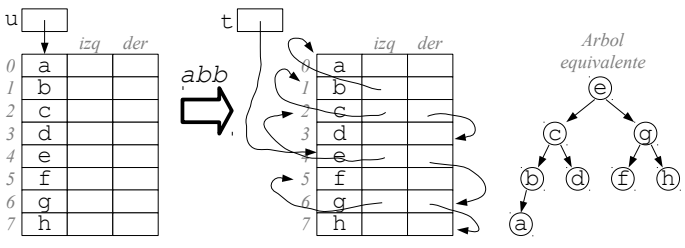
Restricciones: No use el operador de subindicación de arreglos *[ ]* ni su equivalente **(p+i),* use aritmética de punteros. No use *malloc*.

Parte b.- Programe la función *abb* definida como:

```
typedef struct nodo {
  char x;
  struct nodo *izq, *der;
} Nodo;
Nodo *abb(Nodo *u, int n);
```

Esta función arma un árbol de búsqueda binaria a partir de un arreglo *ordenado* de nodos. La figura de más abajo es un ejemplo de uso. A la izquierda se muestra el arreglo *u* de 8 nodos. El tipo de *u* es *Nodo**. Todos los punteros en los campos *izq* y *der* son inicialmente nulos. A la derecha se muestra el resultado de invocar:

```
Nodo* t= abb(u, 0, 7);
```



Restricciones: El tiempo que toma la función *abb* debe ser *O(n)*. Ud. no puede crear nuevos nodos. Reutilice los mismos nodos del arreglo *u*.

Pregunta 2 (30%)

La función *imprimir* sirve para imprimir documentos desde múltiples threads. Recibe el documento que se debe imprimir y un entero entre 0 y 9 que representa la prioridad. La siguiente es una implementación incompleta de esta función porque no considera la prioridad.

```
pthread_mutex_t m= PTHREAD_MUTEX_INITIALIZER;
pthread_cond_t cond= PTHREAD_COND_INITIALIZER;
int ocup= 0;
void imprimir(Doc *doc, int pri) {
  lock(&m);
  while (ocup)
    wait(&cond, &m);
  ocup= 1;
  unlock(&m);

  doPrint(doc); // imprime de verdad

  lock(&m);
  ocup= 0;
  broadcast(&cond);
  unlock(&m);
}
```

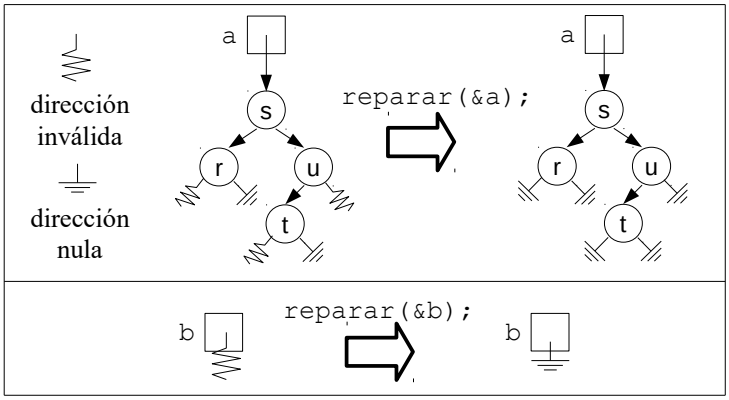
Complete esta implementación de modo que una impresión con prioridad *p* no pueda comenzar mientras exista una impresión pendiente con prioridad *q* > *p*.

Ayuda: Necesitará usar un arreglo global que contabilice las impresiones pendientes para cada prioridad. Programe una función que dada una prioridad *p* retorne verdadero si existen impresiones pendientes con mayor prioridad, o falso en caso contrario. Impresiones con la misma prioridad pueden realizarse en cualquier orden.

Pregunta 3 (30%)

Programe la función: `void reparar(Nodo **pa);`

Esta función debe reparar los punteros que contienen direcciones inválidas en el árbol **pa* reemplazándolos por *NULL*. El tipo *Nodo* es el mismo de la pregunta *1.b*. Una dirección inválida es una dirección que referencia una zona de memoria no asignada al proceso y por lo tanto si el proceso intenta leer o escribir en ella se produce un *segmentation fault* y el proceso recibe la señal SIGSEGV. Las siguientes figuras explican 2 ejemplos de uso. Los punteros *a* y *b* son de tipo *Nodo**.



Ayuda: necesitará capturar la señal SIGSEGV, usar variables globales e invocar *sigsetjmp/siglongjmp*.