Pregunta 1

Parte a.- (2 puntos) El siguiente programa entrega la suma del área de un conjunto de figuras geométricas y el mayor de sus perímetros.

```
typedef struct figura Fig;
struct figura {
  double (*area)(Fig *f);
  double (*perimetro)(Fig *f);
};
typedef struct {
  double (*area)(Fig *f);
  double (*perimetro)(Fig *f);
  double ancho, largo;
} Rect;
```

```
double area_rect(Fig *f) {
  Rect *r= (Rect*)f;
  return r->ancho*r->largo;
}

… completar …
```

```
Rect r1= { area_rect,
  peri_rect, 5., 10. };
Circ c= { … completar …,
  7. };
Rect r2= { area_rect,
  peri_rect, 3., 20.};
Fig *figs[]= { (Fig*)&r1,
  (Fig*)&c, (Fig*)&r2 };
```

```
int main() {
  int i;
  double area_total= 0.;
  double peri_mayor= 0.;
  for (i= 0; i<3; i++) {
    area_total+=
      (*figs[i]->area)(figs[i]);
  }
  printf("El area total es %f\n",
       area_total);
  … completar …
  printf(
    "El mayor perímetro es %f\n",
    peri_mayor);
  return 0;
}
```

Defina el tipo *Circ* y complete las partes indicadas con … en el programa.

Parte b.- (4 puntos) Un archivo contiene un diccionario en el siguiente formato:

alimento:sustancia ingerida por un ser vivo:
casa:edificación construida para ser habitada:
embarcación:todo tipo de artilugio capaz de navegar sobre o bajo el agua:
lluvia:condensación del vapor de agua contenida en las nubes:
… etc ...

El primer carácter ':' separa la palabra de su definición. El segundo ':' termina la definición. Todas las líneas del archivo tienen un número fijo de caracteres para que sea sencillo hacer acceso directo con *fseek*. Las palabras están ordenadas lexicográficamente para que se pueda buscar eficientemente en el archivo mediante búsqueda binaria. Este archivo no cabe en la memoria del computador. Programe la siguiente función:

*char *consultar(char *nom_dic, char *palabra, int n_lin, int ancho);*

En donde *nom_dic* es el nombre del archivo, *palabra* es la palabra que se busca, *n_lin* es el número de líneas del archivo (y por lo tanto el número de palabras y definiciones) y *ancho* es el número de caracteres de cada línea en el archivo. Esta función entrega la definición asociada a *palabra* en el archivo *nom_dic*. Por ejemplo la definición de "alimento" es "sustancia ingerida por un ser vivo". Ud. debe usar búsqueda binaria para encontrar la palabra en el diccionario.

A modo de recuerdo, ésta es la API para manejar archivos:

```
FILE *fopen(char *nom_arch, char *modo);   /* modo== "r" */
size_t fread(char *buf, size_t ancho_item, int n_item, FILE *file);
int fseek(FILE *file, long despl, int w);          /* w==SEEK_SET */
int fclose(FILE *file);
```

Pregunta 2

Los funcionarios pueden asistir libremente a una reunión entrando a la sala en donde se realiza, pero solo pueden salir si todos los asistentes están de acuerdo en concluir la reunión. Los funcionarios son representados por *pthreads*. Programe las siguientes funciones que serán invocadas por los funcionarios:

- `Reunion *nuevaReunion()`: entrega una nueva reunión a la cual los funcionarios pueden asistir.
- `void entrar(Reunion *r)`: ingresa a la reunión *r*.
- `void concluir(Reunion *r)`: vota por finalizar la reunión *r*. Concluir se bloquea en espera hasta que todos los funcionarios que asistieron a la reunión hayan votado por finalizar la reunión.

La figura de al lado muestra con un diagrama de threads un ejemplo de uso de una reunión. Observe que concluir retorna solo cuando los 3 funcionarios que entraron a la reunión votaron por concluirla. Pueden haber distintas reuniones con distintos asistentes y no deben interferir entre sí (es decir no puede usar variables globales en su programación). Resuelva el problema de la sincronización usando un *mutex* y una *condición* por cada reunión.