1) Indicar la salida

```
int main(){
  int *A, *C, *F;
  int **B;
  char *D, *E;
  char G;
  int H;
  H = 68;
  G = 'A';
  A = new int;
  (*A) = H;
  F = &H;
  cout << (*F) << (*A) << endl;;
  B = &F;
  (**B) = 65;
  cout << (*F) << (*A) << (**B) << endl;

  F = new int[3];
  (**B) = 70;

  for (int i = 0; i < 3; i++) {
    F[i] = (*A) + i;
  }
  cout << (*F) << (**B) << endl;

  D = (char*)A;
  E = (char*)(F+1);
  C = (*B);
  C[1] = H;

  cout << (*D) << (*E) << (*C) << endl;
  (*D) = 'B';
  (*E) = 'B';

  for(int i = 0; i < 3; i++) {
    F[i] = 65;
  }
  C = C + 2;
  cout << (*D) << (*E) << (*C) << endl;

  delete A;
  delete[] F;
  return 0;
}
```

2) Implementar para clase Cola las primitivas acolar y desacolar.

3) Implementar la clase BuscadorDeArticulos a partir de las siguientes especificaciones:

```
class BuscadorDeArticulos {
  public:
    /* post: busca en articulos aquellos Artículos que tienen más de comentariosPositivosMinimos comentarios positivos o su
     *       promedio de calificaciones es mayor a calificacionPromedioMinima.
     *       Devuelve una nueva Lista con todos los Artículos en esta condición.
     */
    Lista<Articulo*>* buscarArticulosPopulares(Lista<Articulos*>* articulos,
                                               int comanteriosPositivosMinimos, int calificacionPromedioMinima);
};
```

```
class Articulo {
  public:
    /* post: inicializa el Artículo con el título y contenido
     *       indicados, sin Comentarios ni calificaciones
     *       asociados.
     */
    Articulo(string titulo, string contenido);
    /* post: destruye todos los Comentarios asociados.
     */
    ~Articulo();
    /* post: devuelve el título del Artículo.
     */
    string getTitulo();
    /* post: devuelve el contenido del Artículo
     */
    string getContenido();
    /* post: devuelve los Comentarios asociados al Artículo.
     */
    Lista<Comentario*>* getComentarios();
    /* post: devuelve las calificaciones obtenidas.
     */
    Lista<int>* getCalificaciones();
};
```

```
class Comentario {

  public:
    /* post: inicializa el Comentario con el texto pasado.
     */
    Comentario(string texto, boolean esPositivo);
    /* post: devuelve el texto del Comentario.
     */
    string getTexto();
    /* post: indica si el contenido fue positivo o negativo.
     */
    bool esPositivo();
};
```

4) Diseñar la especificación e implementar los TDAs modelen una **Adivinanza**, cumpliendo las siguientes características:
Una **adivinanza** se crea a partir de un valor numérico comprendido entre 0 y 99 (número a adivinar) y la cantidad de oportunidades para adivinarlo. Se deben proporcionar métodos para:
Arriesgar un valor devolviendo: CORRECTO, MAYOR o MENOR.
Conocer el estado del juego: JUGANDO, GANADO o PERDIDO.
Conocer la cantidad de oportunidades restantes.
Conocer la cantidad de oportunidades usadas.
Conocer la secuencia de valores arriesgados una vez que el juego NO está en estado JUGANDO.

Los alumnos que tienen aprobado el parcialito de punteros no deben realizar el ejercicio 1.
Para aprobar es necesario tener al menos el 60% de los ejercicios correctos y completos.
Para cada método escribir pre y post condición, si recibe argumentos y cuáles, y si retorna un dato y cuál. De faltar ésto, se considerará el código incompleto.
Mantener apagados celulares, i-pod, reproductores de mp3, etc. durante el examen
Duración del examen : 2 horas y 30 minutos