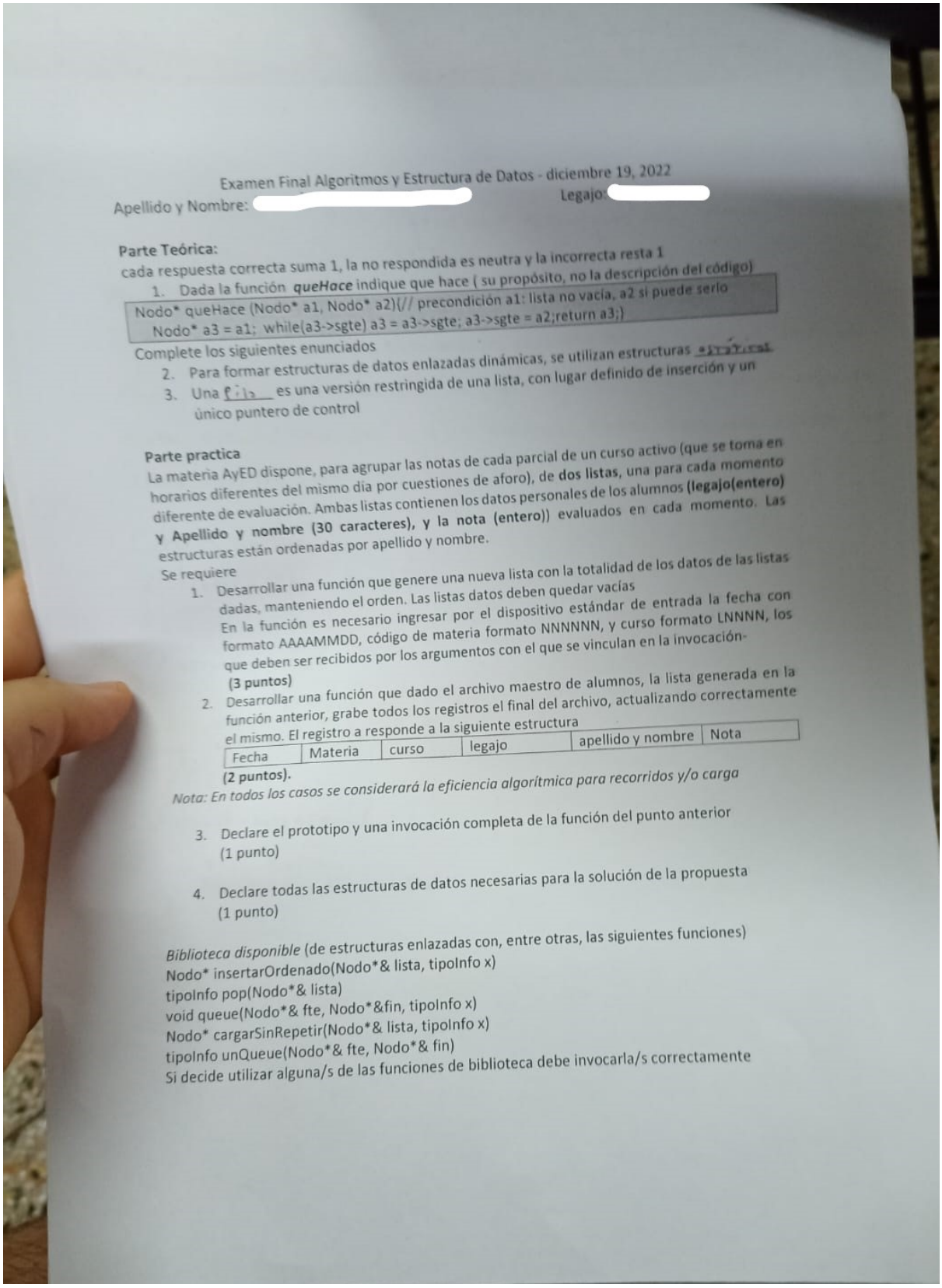
20221912

```
void queHace (Nodo* a1, Nodo* a2)
{Nodo* a3= a1;
while (a3->sgte) //hasta que a3 sea null
{a3=a3->sgte;}
a3->sgte=a2;
return a3;}

Inserta el nodo a2 (y su fila) al final de la lista a1.
```



Examen Final Algoritmos y Estructura de Datos - diciembre 19, 2022

Apellido y Nombre: [redacted] Legajo: [redacted]

Parte Teórica:

cada respuesta correcta suma 1, la no respondida es neutra y la incorrecta resta 1

1. Dada la función ***queHace*** indique que hace (su propósito, no la descripción del código)

```
Nodo* queHace (Nodo* a1, Nodo* a2){// precondición a1: lista no vacía, a2 si puede serlo
  Nodo* a3 = a1;  while(a3->sgte) a3 = a3->sgte; a3->sgte = a2;return a3;}
```

Complete los siguientes enunciados

2. Para formar estructuras de datos enlazadas dinámicas, se utilizan estructuras ejemplos
3. Una Pila es una versión restringida de una lista, con lugar definido de inserción y un único puntero de control

Parte practica

La materia AyED dispone, para agrupar las notas de cada parcial de un curso activo (que se toma en horarios diferentes del mismo día por cuestiones de aforo), de **dos listas**, una para cada momento diferente de evaluación. Ambas listas contienen los datos personales de los alumnos **(legajo(entero) y Apellido y nombre (30 caracteres), y la nota (entero))** evaluados en cada momento. Las estructuras están ordenadas por apellido y nombre.

Se requiere

1. Desarrollar una función que genere una nueva lista con la totalidad de los datos de las listas dadas, manteniendo el orden. Las listas datos deben quedar vacías
   En la función es necesario ingresar por el dispositivo estándar de entrada la fecha con formato AAAAMMDD, código de materia formato NNNNNN, y curso formato LNNNN, los que deben ser recibidos por los argumentos con el que se vinculan en la invocación-
   (3 puntos)
2. Desarrollar una función que dado el archivo maestro de alumnos, la lista generada en la función anterior, grabe todos los registros el final del archivo, actualizando correctamente el mismo. El registro a responde a la siguiente estructura

Fecha	Materia	curso	legajo	apellido y nombre	Nota

   (2 puntos).

Nota: En todos los casos se considerará la eficiencia algorítmica para recorridos y/o carga

3. Declare el prototipo y una invocación completa de la función del punto anterior (1 punto)

4. Declare todas las estructuras de datos necesarias para la solución de la propuesta (1 punto)

Biblioteca disponible (de estructuras enlazadas con, entre otras, las siguientes funciones)

```
Nodo* insertarOrdenado(Nodo*& lista, tipoInfo x)
tipoInfo pop(Nodo*& lista)
void queue(Nodo*& fte, Nodo*&fin, tipoInfo x)
Nodo* cargarSinRepetir(Nodo*& lista, tipoInfo x)
tipoInfo unQueue(Nodo*& fte, Nodo*& fin)
```

Si decide utilizar alguna/s de las funciones de biblioteca debe invocarla/s correctamente

```
struct datosalumno
{int legajo;
 char apellidoynom[30];
 int nota;} //ordenado por apellido y nombre
```

```
struct NodoLista1
{ datoalumno info;
  NodoLista1 sgte;}

struct NodoLista2
{ datoalumno info;
  NodoLista2 sgte;}

struct NodoLista3
{ datoalumno info;
  NodoLista2 sgte;}


//Se quiere agrupar las notas de cada parcial de un curso activo (horarios distintos)
// de dos listas.

1. Desarrollar una función que genere una nueva lista con la totalidad de las listas dadas,
manteniendo orden.
Las listas de datos deben quedar vacias. En la función es necesario
ingresar por dispositivo estandar la fecha en formato AAAAMMDD, codigo de materia en NNNNNNN
y el curso LNNNN. Deben ser recibidos por argumentos con el que se vincula en la invocación.

void nuevalista (NodoLista1* &l, Nodolista2* &l)
{NodoLista3* l3=NULL;
 NodoLista* eliminar1= NULL;
 NodoLista* eliminar2= NULL;
while (l1!=NULL && l2!=NULL)
  {if (strcmp(l1->info.apellidoynom,l2->info.apellidoynom)<0)
    { insertarAlFinal(l3,l1->info);
      eliminar1=l1;
      l1=l1->sgte;
      delete eliminar1;}
   else
    {insertarAlFinal(l3, l2->info);
     eliminar2 = l2;
     l2=l2->sgte;
     delete eliminar2;}
while (l2!=NULL)
{insertarAlFinal(l3, l2->info);
  eliminar2 = l2;
  l2=l2->sgte;
  delete eliminar2;}
while (l1!=NULL)
{ insertarAlFinal(l3,l1->info);
      eliminar1=l1;
      l1=l1->sgte;
      delete eliminar1;}}

2. Desarrollar una función que dada la lista, grabe todos los registros en el final del archivo
actualizando el mismo.

struct registro
{ int fecha;
  int codmateria;
  char curso[5];}

3. Declare el prototipo y una invocación completa de la función anterior.
4. Declare todas las estructuras de datos necesarias para la propuesta.
```