UTN.BA AyED – **Parcial 2** Curso: Día: Turno: Fecha: **-nov-2024**
Apellido: Nombre: Nro.Legajo: Nota:

Enunciado del problema a resolver. (Puntaje Total 7 (siete) sin errores. Tiempo máximo total 1h:30m).

Se evalúan los temas de: Archivos Binarios en C++ stream, Punteros y Estructuras Dinámicas de Datos +Temas Parcial 1.

Se requiere realizar un proceso que "**Informe las respuestas de los alumnos de cálculos de superficies de figuras geométricas**"; contando para ello con los siguientes archivos de datos:

1. **NombresFiguras.Dat**: sin orden, de 20 nombres de figuras geométricas (*str20*).
2. **RespuestasAlumnos.Dat**: ordenado por Nombre del Alumno con repetición, conteniendo cada registro: **2.a)** Nombre Alumno (*str10*), **2.b)** Es Respuesta Correcta (*bool*), **2.c)** Nombre Figura Geométrica (*str20*).

Se deberá considerar el siguiente bloque principal:

```
main() {
 // Definir tipos, variables, inicialización y argumentos a incluir en cada invocación a las funciones.
 Abrir(NombresFiguras,RespAlu);
 ProcNomFig(    );  // Asigna nom. de fig.geom., genera var.din. en c/comp. Genera nodos en Lista. Asigna el valor 0.
 ProcRespAlum(     ); // Debe contar y emitir resp.cor. y resp.incor. usando técnica de Corte de Control, etc.
 EmitirResulta2(    ); // Emitir un informe de cada nom.fig.geom. la cant.de resp.cor. y resp.incor.
 Cerrar(NombresFiguras,RespAlu);
 return 0;
} // main
```

Tabla de diseño de invocaciones y prototipos (interfaz) de funciones

Constantes a utilizar:
const CANT_FIGS = 20;
Completar typedef, struct, arreglos y estructuras dinámicas de datos.
typedef *tListaNomFG;* *sFig* tvrCalc [ ]; str20; str10; tInfo; **struct** sFig sNodo sResp
Prototipos de funciones cuyos módulos deberán desarrollarse e invocar con los argumentos adecuados:
void ProcNomFig(*ifstream* &NomFG, *tvrCalc* vrCalc, *tListaNomFG* &ListaFG);
void ProcRespAlum(*ifstream* &RespAlu, *tListaNomFG* &ListaFG, *tvrCalc* vrCalculos, *ofstream* &ExaAlu);
void EmitirResulta2(*tListaNomFG* &ListaFG, *tvrCalc* vrCalc, *ofstream* &ExaAlu);
Prototipos de funciones que NO deberán desarrollarse pero si llamarlas con los argumentos adecuados:
void OrdxBur(*tvrCalc* vrCalc, ***short*** Card); // **Indicar como comentario el campo por el cual se ordena.**
short BusBinVec(*tvrCalc* vCl, *str20* nomFig, ***short*** ult);
void InsertaInicio(*tListaNomFG* &Lista, *tInfo* valor);
void SacarPrimerNodo(*tListaNomFG* &Lista, *tInfo* &valor);
tListaNomFG BuscarPosNodo(*tListaNomFG* &Lista, ***short*** pos);

UTN.BA AyED – **Parcial 2** Curso: Día: Turno: Fecha: **-nov-2024**
Apellido: Nombre: Nro.Legajo: Nota:

Se pide:

1. Dibujar y codificar los tipos de datos: tListaNomFG, sFig, sNodo, sResp, tvrCalc, tInfo, str20 y str10. **(1 punto).** El dibujo debe ser**: COMPLETO, PROLIJO, SIMPLE, CLARO, y con RÓTULOS APROPIADOS.**

2. Desarrollar las siguientes funciones:
   a) Diagramar la función **ProcNomFig** que asigna los nombres leídos de **NombresFiguras.Dat** a una estructura estática, genere una variable dinámica colgada de la posición de cada figura y asigne el valor de incialización cero. El vector de registro deberá quedar ordenado por el campo nombre de la figura geométrica invocando a ***OrdxBur***. Ademas invocando la función ***InsertaInicio*** debe generar nodos por cada fig.geom. leída en una Lista de Fig.Geom asigna cero al info. **(2 puntos).**
   b) Codificar la función **ProcRespAlum**, que lee los datos de **RespuestasAlumnos.Dat**. Esta función debe procesar los datos del archivo utilizando la técnica de Corte de Control, el cual deberá sumar 1 por cada registro leído, si *respondió bien* en **var. din**. o si *respondió mal* en el **nodo** de la lista de acuerdo en ambos casos, a la fig.geom. correspondiente. Esta función debe invocar a ***BusBinVec*** y a ***BuscarPosNodo***. También deberá emitir títulos apropiados, el nombre del alumno evaluado, la cantidad de respuestas correctas e incorrectas y un mensaje apropiado si la cantidad de resp.cor. es mayor a la cantidad de resp.incor. "Excelente", sino, "Lo Lamento" **(3 puntos).**

Examenes de Superficies de Figuras Geométricas

Alumno/a	cResp.Cor.	cResp.Incor.	Evaluación
X(20)	9	9	Lo lamento
X(20)	9	9	Excelente

   c) Codificar la función **EmiteResulta2** de las cantidades totales de respuestas correctas y de respuestas incorrectas por cada una de las figuruas geométricas. Se debe invocar a la función ***SacarPrimerNodo***. Eliminar todas las variables dinámicas que cuelgan del vector. El diseño del informe es: **(1 punto).**

Listado cant. Resp.Cor. e Incor. por Figuras Geométricas

Nombre Figura	cResp.Cor.	cResp.Incor.
X(20)	99	99
.	.	.

OBSERVACIONES

(1) El enunciado se complementa con explicaciones adicionales y gráficos indicados en la pizarra.
(2) **IMPORTANTE**: Resolver primero el punto 1; tiempo estimado entre 15 y 20 minutos.
(3) Establecer una muestra de datos de los archivos de datos.
(4) Se debe respetar a raja tabla el orden de los campos de cada archivo binario según se indica en esta misma consigna.
(5) Las salidas de los resultados, debe estar dirigida a la consola.
(6) Dado que el **uso de los ciclos afecta el rendimiento del proceso**, se considerará oportuno optimizar estas situaciones de las estructuras de control de programas de la **repetición**, como así también de la **selección** y la **concatenación**. **S.E.ú O.**

UTN.BA AyED – **Parcial 2** Curso: Día: Turno: Fecha: **-nov-2024**
Apellido: Nombre: Nro.Legajo: Nota:

Afirmaciones (Tiempo máximo total 5 min. puntaje máximo 3 (tres) puntos)

Marcar con un círculo la afirmación que ud. considera correcta **V** o **F**, si responde bien, suma 1, y si NO responde, o responde mal, suma 0. **NO escribir en la columna Pntj**.
Realizar una prueba de escritorio o no (usando lápiz y papel), para los ejercicios con código.

	Afirmación	**Marcar**		**Pntj.**

1	Dado el siguiente código: (Puede realizar la Prueba de Escritorio)
	void SacarNodo(*tLista* &Lista, *tInfo* &valor) {
	tLista pElim = Lista;
	
	valor = Lista->info;
	Lista = Lista->sgte;
	delete pElim;
	} // SacarNodo. Elimina el último nodo de la lista.
		V	F	

2	Dado el siguiente código: (Puede realizar la Prueba de Escritorio)
	tLista BuscarPosNodo(*tLista* &Lista, ***short*** pos) {
	tLista pAct = Lista;
	
	for (***short*** i = 1; i <= pos; i++)
	pAct = pAct->sgte;
	return pAct;
	} // BuscarPosNodo
	Recorre la lista pos posiciones y retorna el puntero al nodo apuntado por pAct.
	Es decir, si pos = 0, retorna el puntero al primer nodo de la lista, si pos = 1, el puntero al segundo nodo, en general si pos = n, retorna el puntero al enésimo + 1, nodo de la lista.
		V	F	

3	Dado el siguiente código: f.seekg(-n * sizeof r, ios::cur); Avanza el puntero del archivo f, n componentes desde la posición actual del puntero del archivo f.	V	F	

UTN.BA AyED – **Parcial 2** Curso: Día: Turno: Fecha: **-nov-2024**
Apellido: Nombre: Nro.Legajo: Nota:

Afirmaciones Respuestas (Tiempo máximo total 5 min. puntaje máximo 3 (tres) puntos)

Marcar con un círculo la afirmación que ud. considera correcta **V** o **F**, si responde bien, suma 1, y si NO responde, o responde mal, suma 0. **NO escribir en la columna Pntj**.
Realizar una prueba de escritorio o no (usando lápiz y papel), para los ejercicios con código.

	Afirmación	**Marcar**		**Pntj.**

1	Dado el siguiente código: (Puede realizar la Prueba de Escritorio)
	void SacarNodo(*tLista* &Lista, *tInfo* &valor) {
	tLista pElim = Lista;
	
	valor = Lista->info;
	Lista = Lista->sgte;
	delete pElim;
	} // SacarNodo. Elimina el último nodo de la lista.
	FALSO
	Copia al puntero pElim la dirección donde apunta Lista, que apunta el primer nodo de la lista, luego, al hacer **delete pElim**, elimina **NO el último nodo** sino el **primer nodo** de la lista.
		V	**F**	

2	Dado el siguiente código: (Puede realizar la Prueba de Escritorio)
	tLista BuscarPosNodo(*tLista* &Lista, ***short*** pos) {
	tLista pAct = Lista;
	
	for (***short*** i = 1; i <= pos; i++)
	pAct = pAct->sgte;
	return pAct;
	} // BuscarPosNodo
	Recorre la lista pos posiciones y retorna el puntero al nodo apuntado por pAct.
	Es decir, si pos = 0, retorna el puntero al primer nodo de la lista, si pos = 1, el puntero al segundo nodo, en general si pos = n, retorna el puntero al enésimo + 1, nodo de la lista.
	VERDADERO
	Si pos es cero, no ingresa al ciclo exacto y la fución retorna pAct, que fuera inicializado con Lista, por lo que retorna el puntero al primer nodo de la Lista, si por es uno, ingresa una vez al ciclo, avanzando pAct al siguiente nodo, al salir del ciclo retorna la dirección al segundo nodo, y asi sucesivamente para pos mayor a uno, retornando la dirección del nodo pos + 1.
		V	F	

3	Dado el siguiente código: f.seekg(-n * sizeof r, ios::cur); Avanza el puntero del archivo f, n componentes desde la posición actual del puntero del archivo f. FALSO Al ser n negativo luego de aplicar el operador monario menos, el puntero al archivo debe **RETROCEDER** desde la posición actual n componentes previas.	V	**F**	

Folleto explicativo del problema a resolver

ESTRUCTURAS DE DATOS

#define record **struct**

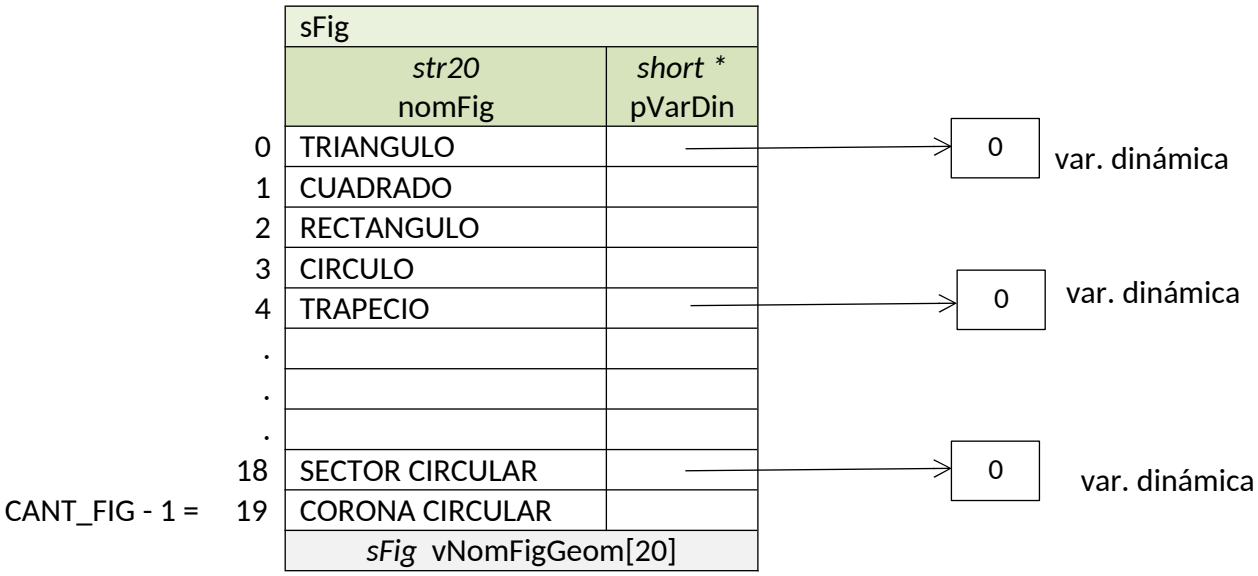
Archivo: NombresFiguras.Dat

Contiene los nombres de 20 figuras geométricas desordenados.

Archivo: RespuestasAlumnos.Dat

Se encuentra ordenado por nomAlu con repetición. El campo *EsRespCor* contiene el valor cero (respondió mal) o el valor 1 (respondió bien).

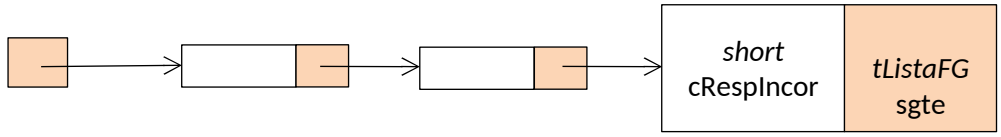
sResp		
str10 nomAlu 11b.	*bool* EsRespCor 1b.	*str20* nomFig 21b.



Primer instancia: Recorriendo secuencialmente el archivo RespuestasAlumnos.Dat se guarda en el campo nomFig el nombre de la figura leída, se genera una nueva variable dinámica de tipo short ambos datos colgados en la posición i del vector vNomFigGeom, con i igual al valor cero, que se incrementará por cada nuevo registro leído del archivo mencionado. Esto se repite 20 veces, una por cada figura geométrica leída. En forma simultánea con el ciclo se generan 20 nodos en la lista ListaFG insertando por inicio invocando al módulo *InsertaInicio*, asignando al campo cRespIncor el valor cero.

Segunda Instancia: Al salir del ciclo, se ordena la tabla por el campo nomFig invocando al módulo *OrdxBur*.

Los valores almacenados en cada una de las variables dinámicas, representa la **cantidad de respuestas correctas** de cada figura geométrica.



tListaFG ListaFG

Los valores almacenados en cada uno de los nodos, representa la **cantidad de respuestas incorrectas** de cada figura geométrica. Cada nodo representa una figura geométrica que se corresponde con cada una de las posiciones en el vector de nombres de figuras ya ordenado. El nodo que representa la figura CIRCULO es el primero, el nodo que representa la figura TRIANGULO es el último.

Al procesar las respuestas de los alumnos correspondiente a una figura geométrica, primero, se busca en el vector esa figura, encontrado, se obtiene la posición en el vector, luego, se recorre la lista ListaFG hasta alcancar el nodo que se corresponde con la posición ordinal obtenida del vector.