**1ER PARCIAL – ESQUEMA**

**(Puede variar algo, aunque siempre estará dentro de los conceptos y herramientas vistas)**

|  |  |  |
| --- | --- | --- |
| **Tema** | **Forma de evaluación** | **Puntaje** |
| **Conceptos básicos:**  Algoritmos, variable, etc.  **Comandos:**  Asignación externa de Entrada  Asignación externa de Salida  Asignación interna  Alternativa  Alternativa múltiple  Ciclo exacto  Ciclo precondicional  Ciclo postcondicional  **Funciones:**  void  no void  **Estructuras de datos:**  Variables simples  Arreglos  Registros  Archivos de texto y binarios | **Preguntas tipo múltiple choice (entre 2 y 3 preguntas)**  Ejemplo/s (la/las del parcial será/n otra/s)  **¿Cuál es la diferencia en C++ entre un ciclo precondicional y un ciclo postcondicional?**   1. que en uno se pone la condición de corte y en el otro la de continuidad 2. que en el precondicional la condición puede ser compuesta y en el postcondicional no 3. que en el ciclo postcondicional las acciones en el ciclo se ejecutan al menos una vez 4. que en el ciclo postcondicional las acciones en el ciclo pueden no ejecutarse ni siquiera una vez   **¿Cuál es la diferencia entre los parámetros en una función void y una no void?**   1. que la void retorna un valor en su nombre 2. que los parámetros en una no void están limitados en número 3. que la void siempre tendrá al menos un parámetro de ida y vuelta (referencia) al no retornar nada en el nombre 4. que al invocar una no void se debe realizar alguna acción con el valor de retorno   Recordemos por qué la d  ej. función no void  float Promedio (int cant; int total) {  Return total / (cant \* 1.0);  }  Al invocarla debo realizar alguna acción con la invocación, por ejemplo asignar su valor a una var  ..  p = Promedio (n,tot) ;  .. | 2 pts |
| **TODOS**  Variables simples  Arreglos  Registros  Archivos de texto y binarios | **Seguimiento de Ejercicio**  **Qué hace / Qué devuelve / Qué salida de datos produce**  Ejemplo (el del parcial será otro)  **¿Qué hace el sgte algoritmo? O ¿Qué salida de datos produce?**  (les daré un pdf con el código a seguir, y asimismo el diagrama correspondiente para quienes lo ven mejor en diagrama)  **int** **main**() {  **int** c,n,nro,m;  **int** vec[5];  **for** (c=0;c<=5;c++) {  vec[c] = 0;  };  cout << "Ingrese cantidad " << **endl**;  cin >> n;  **for** (c=1;c<=n;c++) {  cout << "Ing nro entre (0 a 4)" << **endl**;  cin >> nro;  vec[nro]++;  };  **for** (c=0;c<=4;c++) {  cout << c << “ “ << vec[c] << endl;  **if** ((c==0) || (vec[c]>m)) {  m = vec[c];  }  };  cout << m << **endl**;  }  **Si se pide ¿Qué salida de datos produce?**  **Daría por ejemplo los sgtes datos de entrada para seguimiento**  n: 8  nros: 3, 4, 2, 1, 3, 4, 0, 3  **(Cálculo interno para seguimiento que seguramente harían en sus cuaderno para ir simulando el comportamiento del vector)**   |  |  | | --- | --- | | 0 | 1 | | 1 | 1 | | 2 | 1 | | 3 | 3 | | 4 | 2 |   **Las salidas de datos serán**  0 1  1 1  2 1  3 3  4 2  3  **Si se pide** **¿Qué hace el sgte algoritmo?**  **Se darían opciones para elegir la correcta, por ejemplo:**  -Ingresa n valores entre 0 y 4 hasta que se ingresa un cero y cuenta la cantidad de veces que se ingresa cada número.  -Ingresa n valores entre 0 y 4 hasta que se ingresa un cero y cuenta la cantidad de veces que se ingresa cada número si es mayor que el otro.  -Ingresa n valores entre 0 y 4 y cuenta la cantidad de veces que se ingresa cada número, siempre que sea mayor que el otro.  -Ingresa n valores entre 0 y 4 y cuenta la cantidad de veces que se ingresa cada número, y busca luego el nro que tuvo mayor cantidad de ingresos.  -Ingresa n valores entre 0 y 4 y cuenta la cantidad de veces que se ingresa cada número, y busca luego el nro con mayor cantidad de ingresos si es igual a cero.  -Ingresa n valores entre 0 y 4 y acumula en un vector los números ingresados, y busca luego el máximo.  -Ingresa n valores entre 0 y 4 y acumula en un vector los números ingresados, y busca luego el máximo o el que sea igual a cero.  -Ingresa n valores entre 0 y 4 y acumula en un vector los números ingresados, y busca luego el mínimo.  -Ingresa n valores entre 0 y 4 y acumula en un vector los números ingresados, y busca luego el mínimo o el que sea igual a cero. | 2 pts |
| **TODOS**  Variables simples  Arreglos  Registros  Archivos de texto y binarios | **Desarrollo de ejercicio modular**  Ejemplo (el del parcial será otro)  -Una empresa de venta de productos de cosmética. Se cuenta con un archivo Productos.dat con los siguientes campos CodP entero y PU decimal.  Desarrollar un módulo **CargayBusca** que recibe como parámetro un dato de “cod producto” a buscar, *carga la información del archivo de los productos que cuestan más que 100 a un vector*, y *busca entre ellos dicho CodP pudiendo estar o no*. Debe devolver la pos en que lo encontró o -1 en caso de no haberlo encontrado.  Agréguele al módulo todos los parámetros que requiera.  Escriba la línea (es una sola) que invoca a dicho módulo en el main sin necesidad de declarar ni dar valor a los parámetros que invoca  void **CargayBusca**(file\* aProds, tprod vec, int & tope, int Codp, int & pos) //aProds puede no ponerse  como parámetro y trabajarse localmente como a continuación  FILE \*aProds = fopen ("Productos.dat", "rb");  tprod Rprod;  **int** i=0;  fread (&RProd, **sizeof** (RProd), 1, aProds); // read (RProd, aProds)  **while** (!feof (aProds)){  if (RProd.PU > 100) {  VProd[i]=RProd;  i++;  };  fread (&RProd, **sizeof** (RProd), 1, aProds); // read(RProd,aProds)  };  tope=i-1;  fclose (aProds);  i=0;  **while**((i<=tope) && (VProd[i].Codp!=Codp)){  i++;  };  **if** (i>tope){  pos = -1;  }  **else** {  pos = i;  }  };  Como **no void** era igual pero no llevaba el parámetro pos y al final hubiese llevado **Return pos**  int **CargayBusca**(file\* aProds, tprod vec, int & tope, int Codp) //aProds puede no ponerse el archivo  como parámetro y trabajarlo localmente  ..  ..  Return pos;  };  **Invocación en el main**  ..  **CargayBusca**(aProds, vec, tope, Codp, pos); // como void  ..  ..  **pos = CargayBusca**(aProds, vec, tope, Codp); // como no void  .. | 4 pts |
| **TODOS**  Variables simples  Arreglos  Registros  Archivos de texto y binarios | **Desarrollo ejercicio modular**  -Enunciado  Desarrollar un módulo XXX que recibe zzz y que debe hacer tal cosa y devolver tal otra. (va a ser un problema que no sea para nada dentro de los módulos estándar vistos) | 2 pts  (para  posibilidad de promoción) |