



UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA
Facultad de arquitectura, ingeniería y diseño.(FIAD).

Actividad 9: librerías en c, métodos de ordenación y búsqueda.

Alumna: Luisa Lizeth Zerega Soto.

Materia: Laboratorio de Programación Estructurada.

Clave: 36276.

Fecha de entrega:

Matricula: 356491.

Maestro: Pedro Nunez Yepiz .

Tema-Unidad: librerías en c, métodos de ordenación y búsqueda

Introducción.

Las **bibliotecas** contienen variables y funciones dentro de ellas. Algunos tipos de archivos que podemos importar o incluir en nuestro programa se conocen como bibliotecas (o librerías).

Estos archivos contienen las especificaciones de varios recursos ya contruidos y utilizables, como leer el teclado o mostrar algo en la pantalla, entre muchos otros.

Al poder incluir estas bibliotecas con definiciones de diferentes funcionalidades podemos ahorrar muchas cosas. Echemos un vistazo al contenido de las bibliotecas más utilizadas.

La **búsqueda** permite encontrar un elemento concreto del conjunto, mientras que el **ordenamiento** consiste en localizar los datos según un criterio para facilitar la búsqueda del elemento buscado o identificar las relaciones entre los datos.

Competencia.

El alumno será capaz de realizar programas en c de tipo menú que contengan el correcto funcionamiento de librerías en c, métodos de ordenación y búsqueda.

Fundamentos.

- Google Drive: Sign-in. (n.d.-a).
https://drive.google.com/drive/folders/1zfCEZoq73gj1LtTC7koNSF6AILzfJ_9p
- Google Drive: Sign-in. (n.d.-b).
<https://drive.google.com/drive/folders/16b9iTpAtErHrvXoiByItnFVFOWFpbtWI>
- Google Drive: Sign-in. (n.d.-c).
<https://drive.google.com/drive/folders/16b9iTpAtErHrvXoiByItnFVFOWFpbtWI>

Procedimiento.

Realiza programa en C utilizando librería propia, el programa deberá tener el siguiente menú.

MENÚ

- 1.- LLENAR VECTOR
- 2.- LLENAR MATRIZ
- 3.- IMPRIMIR VECTOR
- 4.- IMPRIMIR MATRIZ
- 5.- ORDENAR VECTOR
- 6.- BUSCAR VALOR EN VECTOR
- o.- SALIR

NOTA: El programa deberá repetirse cuantas veces lo desee el usuario,
Validado el menú con la función vali_num

INSTRUCCIONES

- 1.- LLENAR VECTOR .- Llenar vector con 15 números, los números generados aleatoriamente, los números entre el rango de 100 al 200 (no repetidos)
- 2.- LLENAR MATRIZ .- Llenar la matriz de 4x4 con con números generados aleatoriamente, números entre el rango de 1 al 16 (no repetidos)
- 3.- IMPRIMIR VECTOR .- Imprime el vector que se envíe, donde la función recibe como parámetro el vector,tamaño, nombre del vector.
- 4.- IMPRIMIR MATRIZ.- Imprime la matriz sin importar el tamaño de la matriz recibiendo como parámetros la matriz, la cantidad de renglones y columnas, así como nombre que se le dará a la matriz
- 5.- ORDENAR VECTOR.- Usar función que ordene el vector por el método de ordenación de la Burbuja mejorada.
- 6.- BUSCAR VALOR EN VECTOR.- Buscar un valor en el vector usando el método de búsqueda secuencial.
- o.- SALIR

Resultados.

[illegible][illegible]

```

1  #include <iostream>
2  #include <vector>
3  #include <string>
4  #include <string_view>
5  #include <algorithm>
6  #include <limits>
7
8  using namespace std;
9
10 void menu() {
11     cout << "Menu de opciones:\n";
12     cout << "1. Agregar elemento\n";
13     cout << "2. Eliminar elemento\n";
14     cout << "3. Buscar elemento\n";
15     cout << "4. Imprimir vector\n";
16     cout << "5. Salir\n";
17 }
18
19 int main() {
20     vector<string> v;
21     int opcion;
22     while (opcion != 5) {
23         menu();
24         cout << "Ingrese una opcion: ";
25         int opcion;
26         while (opcion < 1 || opcion > 5) {
27             cout << "Opcion invalida. Por favor, ingrese una opcion valida.\n";
28             continue;
29         }
30         switch (opcion) {
31             case 1: {
32                 string s;
33                 cout << "Ingrese un string: ";
34                 getline(cin, s);
35                 v.push_back(s);
36                 break;
37             }
38             case 2: {
39                 int i;
40                 cout << "Ingrese el indice del elemento a eliminar: ";
41                 while (i < 0 || i > v.size() - 1) {
42                     cout << "Indice invalido. Por favor, ingrese un indice valido.\n";
43                     continue;
44                 }
45                 v.erase(v.begin() + i);
46                 break;
47             }
48             case 3: {
49                 string s;
50                 cout << "Ingrese un string: ";
51                 getline(cin, s);
52                 auto it = find(v.begin(), v.end(), s);
53                 if (it != v.end()) {
54                     cout << "El string " << s << " se encuentra en el vector.\n";
55                 } else {
56                     cout << "El string " << s << " no se encuentra en el vector.\n";
57                 }
58                 break;
59             }
60             case 4: {
61                 cout << "El vector contiene los siguientes elementos:\n";
62                 for (const auto& s : v) {
63                     cout << s << " ";
64                 }
65                 cout << "\n";
66                 break;
67             }
68             case 5: {
69                 return 0;
70             }
71         }
72     }
73     return 0;
74 }

```

The screenshot displays a Windows 11 desktop environment. The primary focus is the Visual Studio Code (VS Code) application, which is open to a C++ source file named 'Estructura.cpp'. The code defines a 10x10 matrix 'A' and a vector 'v', both initialized with random values. The program then prints the elements of the vector 'v'. The VS Code interface includes a sidebar on the left with a file explorer showing the project structure, a central editor pane with the code, and a bottom status bar indicating the current file and line numbers.

The code in the editor is as follows:

```

1 // Estructura.cpp
2
3 #include <iostream>
4 #include <vector>
5 #include <random>
6
7 using namespace std;
8
9 // Función para generar un vector de tamaño 'n' con valores aleatorios entre 0 y 100.
10 void llenarVector(int vector[], int tam) {
11     srand(time(NULL));
12
13     for (int i = 0; i < tam; i++) {
14         vector[i] = rand() % 101 + 100;
15     }
16
17     print("vector llenado exitosamente.\n");
18 }
19
20 // Función para generar una matriz de tamaño 'n' x 'm' con valores aleatorios entre 0 y 100.
21 void llenarMatriz(int matriz[][10]) {
22     srand(time(NULL));
23
24     int numeros[10] = {0};
25     int num;
26     int fila, columna;
27
28     for (int i = 0; i < 10; i++) {
29         do {
30             num = rand() % 101 + 1;
31         } while (numeros[num - 1] != 0);
32
33         numeros[num - 1] = 1;
34         fila = i / 4;
35         columna = i % 4;
36         matriz[fila][columna] = num;
37     }
38 }

```

The taskbar at the bottom of the screen shows the Start button, task view, and several open applications: Edge, File Explorer, VS Code, and a terminal window. The system tray on the right shows the date and time as 10/14/2023, 10:42 AM.

[illegible]

The screenshot shows a C++ IDE with a project named "ZSLASCRDP". The file explorer on the left lists the project files, including "ZSLASCRDP.cpp", "ZSLASCRDP.h", "ZSLASCRDP.vcxproj", "ZSLASCRDP.vcxproj.filters", and "ZSLASCRDP.sln". The code editor on the right displays the following C++ code:

```

101 // Dot product
102 // Autor: Víctor Manuel Rodríguez
103 // Fecha: 10/05/2023
104 // Descripción: Cálculo del producto punto de dos vectores
105 // Versión: 1.0
106 //
107 //
108 //
109 //
110 //
111 //
112 //
113 //
114 //
115 //
116 //
117 //
118 //
119 //
120 //
121 //
122 //
123 //
124 //
125 //
126 //
127 //
128 //
129 //
130 //
131 //
132 //
133 //
134 //
135 //
136 //
137 //
138 //
139 //
140 //
141 //
142 //
143 //
144 //
145 //
146 //
147 //
148 //
149 //
150 //
151 //
152 //
153 //
154 //
155 //
156 //
157 //
158 //
159 //
160 //
161 //
162 //
163 //
164 //
165 //
166 //
167 //
168 //
169 //
170 //
171 //
172 //
173 //
174 //
175 //
176 //
177 //
178 //
179 //
180 //
181 //
182 //
183 //
184 //
185 //
186 //
187 //
188 //
189 //
190 //
191 //
192 //
193 //
194 //
195 //
196 //
197 //
198 //
199 //
200 //
201 //
202 //
203 //
204 //
205 //
206 //
207 //
208 //
209 //
210 //
211 //
212 //
213 //
214 //
215 //
216 //
217 //
218 //
219 //
220 //
221 //
222 //
223 //
224 //
225 //
226 //
227 //
228 //
229 //
230 //
231 //
232 //
233 //
234 //
235 //
236 //
237 //
238 //
239 //
240 //
241 //
242 //
243 //
244 //
245 //
246 //
247 //
248 //
249 //
250 //
251 //
252 //
253 //
254 //
255 //
256 //
257 //
258 //
259 //
260 //
261 //
262 //
263 //
264 //
265 //
266 //
267 //
268 //
269 //
270 //
271 //
272 //
273 //
274 //
275 //
276 //
277 //
278 //
279 //
280 //
281 //
282 //
283 //
284 //
285 //
286 //
287 //
288 //
289 //
290 //
291 //
292 //
293 //
294 //
295 //
296 //
297 //
298 //
299 //
300 //
301 //
302 //
303 //
304 //
305 //
306 //
307 //
308 //
309 //
310 //
311 //
312 //
313 //
314 //
315 //
316 //
317 //
318 //
319 //
320 //
321 //
322 //
323 //
324 //
325 //
326 //
327 //
328 //
329 //
330 //
331 //
332 //
333 //
334 //
335 //
336 //
337 //
338 //
339 //
340 //
341 //
342 //
343 //
344 //
345 //
346 //
347 //
348 //
349 //
350 //
351 //
352 //
353 //
354 //
355 //
356 //
357 //
358 //
359 //
360 //
361 //
362 //
363 //
364 //
365 //
366 //
367 //
368 //
369 //
370 //
371 //
372 //
373 //
374 //
375 //
376 //
377 //
378 //
379 //
380 //
381 //
382 //
383 //
384 //
385 //
386 //
387 //
388 //
389 //
390 //
391 //
392 //
393 //
394 //
395 //
396 //
397 //
398 //
399 //
400 //
401 //
402 //
403 //
404 //
405 //
406 //
407 //
408 //
409 //
410 //
411 //
412 //
413 //
414 //
415 //
416 //
417 //
418 //
419 //
420 //
421 //
422 //
423 //
424 //
425 //
426 //
427 //
428 //
429 //
430 //
431 //
432 //
433 //
434 //
435 //
436 //
437 //
438 //
439 //
440 //
441 //
442 //
443 //
444 //
445 //
446 //
447 //
448 //
449 //
450 //
451 //
452 //
453 //
454 //
455 //
456 //
457 //
458 //
459 //
460 //
461 //
462 //
463 //
464 //
465 //
466 //
467 //
468 //
469 //
470 //
471 //
472 //
473 //
474 //
475 //
476 //
477 //
478 //
479 //
480 //
481 //
482 //
483 //
484 //
485 //
486 //
487 //
488 //
489 //
490 //
491 //
492 //
493 //
494 //
495 //
496 //
497 //
498 //
499 //
500 //
501 //
502 //
503 //
504 //
505 //
506 //
507 //
508 //
509 //
510 //
511 //
512 //
513 //
514 //
515 //
516 //
517 //
518 //
519 //
520 //
521 //
522 //
523 //
524 //
525 //
526 //
527 //
528 //
529 //
530 //
531 //
532 //
533 //
534 //
535 //
536 //
537 //
538 //
539 //
540 //
541 //
542 //
543 //
544 //
545 //
546 //
547 //
548 //
549 //
550 //
551 //
552 //
553 //
554 //
555 //
556 //
557 //
558 //
559 //
560 //
561 //
562 //
563 //
564 //
565 //
566 //
567 //
568 //
569 //
570 //
571 //
572 //
573 //
574 //
575 //
576 //
577 //
578 //
579 //
580 //
581 //
582 //
583 //
584 //
585 //
586 //
587 //
588 //
589 //
590 //
591 //
592 //
593 //
594 //
595 //
596 //
597 //
598 //
599 //
600 //
601 //
602 //
603 //
604 //
605 //
606 //
607 //
608 //
609 //
610 //
611 //
612 //
613 //
614 //
615 //
616 //
617 //
618 //
619 //
620 //
621 //
622 //
623 //
624 //
625 //
626 //
627 //
628 //
629 //
630 //
631 //
632 //
633 //
634 //
635 //
636 //
637 //
638 //
639 //
640 //
641 //
642 //
643 //
644 //
645 //
646 //
647 //
648 //
649 //
650 //
651 //
652 //
653 //
654 //
655 //
656 //
657 //
658 //
659 //
660 //
661 //
662 //
663 //
664 //
665 //
666 //
667 //
668 //
669 //
670 //
671 //
672 //
673 //
674 //
675 //
676 //
677 //
678 //
679 //
680 //
681 //
682 //
683 //
684 //
685 //
686 //
687 //
688 //
689 //
690 //
691 //
692 //
693 //
694 //
695 //
696 //
697 //
698 //
699 //
700 //
701 //
702 //
703 //
704 //
705 //
706 //
707 //
708 //
709 //
710 //
711 //
712 //
713 //
714 //
715 //
716 //
717 //
718 //
719 //
720 //
721 //
722 //
723 //
724 //
725 //
726 //
727 //
728 //
729 //
730 //
731 //
732 //
733 //
734 //
735 //
736 //
737 //
738 //
739 //
740 //
741 //
742 //
743 //
744 //
745 //
746 //
747 //
748 //
749 //
750 //
751 //
752 //
753 //
754 //
755 //
756 //
757 //
758 //
759 //
760 //
761 //
762 //
763 //
764 //
765 //
766 //
767 //
768 //
769 //
770 //
771 //
772 //
773 //
774 //
775 //
776 //
777 //
778 //
779 //
780 //
781 //
782 //
783 //
784 //
785 //
786 //
787 //
788 //
789 //
790 //
791 //
792 //
793 //
794 //
795 //
796 //
797 //
798 //
799 //
800 //
801 //
802 //
803 //
804 //
805 //
806 //
807 //
808 //
809 //
810 //
811 //
812 //
813 //
814 //
815 //
816 //
817 //
818 //
819 //
820 //
821 //
822 //
823 //
824 //
825 //
826 //
827 //
828 //
829 //
830 //
831 //
832 //
833 //
834 //
835 //
836 //
837 //
838 //
839 //
840 //
841 //
842 //
843 //
844 //
845 //
846 //
847 //
848 //
849 //
850 //
851 //
852 //
853 //
854 //
855 //
856 //
857 //
858 //
859 //
860 //
861 //
862 //
863 //
864 //
865 //
866 //
867 //
868 //
869 //
870 //
871 //
872 //

```

The screenshot shows a C++ IDE with the following code:

```

1  #include <iostream>
2  #include <vector>
3  #include <string>
4  #include <algorithm>
5
6  using namespace std;
7
8  struct Actividad {
9      string nombre;
10     int duracion;
11 };
12
13 void ordenarVector(vector<Actividad> &v, int tam) {
14     for (int i = 0; i < tam - 1; i++) {
15         for (int j = i + 1; j < tam; j++) {
16             if (v[i].duracion > v[j].duracion) {
17                 Actividad temp = v[i];
18                 v[i] = v[j];
19                 v[j] = temp;
20             }
21         }
22     }
23     cout << "Vector ordenado exitosamente.\n";
24 }
25
26 void buscarValor(vector<Actividad> &v, int tam, int valor) {
27     for (int i = 0; i < tam; i++) {
28         if (v[i].duracion == valor) {
29             cout << "El valor " << valor << " se encuentra en la posición " << i << " del vector.\n";
30             return i;
31         }
32     }
33     cout << "El valor " << valor << " no se encuentra en el vector.\n";
34     return -1;
35 }
36
37 int main() {
38     vector<Actividad> v;
39     v.push_back(Actividad("compartir", 11));
40     v.push_back(Actividad("hacerpaz", 10));
41     v.push_back(Actividad("batarea", 5));
42     v.push_back(Actividad("hacerpaz", 5));
43
44     ordenarVector(v, v.size());
45
46     int valor;
47     cout << "Ingrese un valor para buscar: ";
48     cin >> valor;
49
50     int pos = buscarValor(v, v.size(), valor);
51     if (pos != -1) {
52         cout << "El valor " << valor << " se encuentra en la posición " << pos << " del vector.\n";
53     } else {
54         cout << "El valor " << valor << " no se encuentra en el vector.\n";
55     }
56
57     return 0;
58 }

```

The output of the program is:

```

compartir 11min
hacerpaz 10min
batarea 5min
hacerpaz 5min

```

Conclusiones.

Se fue capaz de realizar programas en c de tipo menú que contengan el correcto funcionamiento de librerías en c, métodos de ordenación y búsqueda.

Anexos.

