**Звіт**

До лабораторної роботи №1

Алгоритми і структури даних

Студента ІПЗ-11/2

Майбороди Ігоря

1. **Умова задачі**

Написати програму мовою C# з можливістю вибору різних алгоритмів пошуку. Продемонструвати роботу (ефективність, час виконання) програм на різних структурах даних (масив, лінійний зв’язаний список), з різними умовами, що забезпечують зменшення часу виконання. Навести аналіз отриманих результатів.

Реалізувати алгоритми:

• пошуку перебором елемента масиву, що дорівнює заданому значенню.

• пошуку з бар'єром елемента масиву, що дорівнює заданому значенню.

• бінарного пошуку елемента масиву рівного заданому значенню.

• бінарного пошуку елемента масиву, рівного заданому значенню, в якій нове значення індексу m визначалося б не як середнє значення між L і R, а згідно з правилом золотого перерізу.

1. **Аналіз задачі**

Я реалізував 4 різні алгоритми пошуку елемента (цілого числа) в масиві та в лінійному списку. Загалом отримав 8 різних методів в класі Program, які відповідають за окремий алгоритм певних структур даних. Додатково був створений клас Node з полями data (дані) та next (посилання на наступний об’єкт класу) для реалізації лінійного списку.

На початку користувач обирає діапазон значень чисел (найменше та найбільше), які будуть в масиві чи в Linked list. Потім з головної функції Main звертаємося до методу Menu, у якій ми можемо обрати структуру даних (масив чи список) з якою будемо працювати, а потім за допомогою оператору swith обираємо алгоритм пошуку. Змінювати алгоритм пошуку користувач може до того моменту, доки не введе 0.

У кожному алгоритмі буде створюватися кожний раз новий масив чи список з кількістю, яку введе користувач. Для демонстрації правильності роботи алгоритмів на консоль будуть виводитися всі значення елементів масиву чи linked list.

Що стосується алгоритмів бінарного пошуку та бінарного пошуку за правилом золотого перерізу, то масив чи список буде додатково сортуватися за зростанням. Для перевірки ефективності виконання кожного алгоритму користувач отримає на консолі значення часу виконання алгоритму в мілісекундах, це можливо знайти використовуючи клас Stopwatch бібліотеки System.Diagnostics.

1. **Структура основних вхідних та вихідних даних**

Вхідні дані (цілі числа): мінімальне і максимальне число діапазону значень; кількість елементів масиву чи списку; число, яке хочу знайти.

Вихідні дані (цілі числа): масив та список рандомних чисел; індекс елемента, якщо він знаходиться в масиві чи списку.

1. **Алгоритм розв'язання задачі**

*Структура даних – масив:*

1) Лінійний пошук

//алгоритм лінійного пошуку

bool found = false;

while (j < size && !found)

{

if (arr[j] == key)

{

index = j;

found = true;

}

else j++;

}

if (!found)

{

Console.WriteLine("Element not exist!");

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

else

{

Console.WriteLine("Element exist! Index = {0}", index);

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

2) Лінійний пошук з бар'єром

//алгоритм лінійного пошуку з бар'єром

arr[size] = key; //додаємо в кінець елемент масиву, який = шуканому

while (arr[j] != key)

{

j++;

}

if (j == size)

{

Console.WriteLine("Element not exist!");

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

else

{

Console.WriteLine("Element exist! Index = {0}", j);

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

3) Бінарний пошук

//бінарний пошук

int left = 0, right = size - 1, middle = 0;

bool found = false;

while (left <= right && !found)

{

middle = (left + right) / 2;

if (arr[middle] == key) found = true;

else if (arr[middle] > key) right = middle - 1;

else left = middle + 1;

}

if (!found)

{

Console.WriteLine("Element not exist!");

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

else

{

Console.WriteLine("Element exist! Index = {0}", middle);

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

4) Бінарний пошук за правилом золотого перерізу

//бінарний пошук удосконалений

int left = 0, right = size - 1, middle = 0;

bool found = false;

while (left < right && !found)

{

middle = Convert.ToInt32(right - (right - left) / 1.618);

if (arr[middle] == key) found = true;

else if (arr[middle] > key) right = middle - 1;

else left = middle + 1;

}

if (!found)

{

Console.WriteLine("Element not exist!");

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

else

{

Console.WriteLine("Element exist! Index = {0}", middle);

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

*Структура даних – linked list:*

1) Лінійний пошук

if (search(head, key) >= 0)

{

Console.WriteLine("Index of element = " + (search(head, key) - 1));

sWatch.Stop();

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

else

{

Console.WriteLine("Not exist");

sWatch.Stop();

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

Допоміжний метод для пошуку елемента в списку

static int search(Node head, int key)

{

int x = 0;

while (head != null)

{

x++;

if (head.data == key)

return x;

head = head.next;

}

return -1;

}

2) Лінійний пошук з бар'єром

head.AddToEnd(key); //додаємо число в кінець списку

while (head.data != key)

{

head = head.next;

x++;

}

if (x != size)

{

Console.WriteLine("Index of element = " + (x));

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

else

{

Console.WriteLine("Not exist");

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

3) Бінарний пошук

//бінарний пошук

int center, right = size, left = 0;

do

{

center = right - ((right - left) / 2);

if (GetElement(head, center) < key) left = center + 1;

else right = center - 1;

} while ((left <= right) && (GetElement(head, center) != key));

if (GetElement(head, center) == key)

{

Console.WriteLine("Index of your element " + (center - 1));

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

else

{

Console.WriteLine("Key isn't included in array");

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

Допоміжний метод для отримання елемента в списку

static int GetElement(Node head, int position)

{

for (int i = 0; i < position - 1; i++)

{

head = head.next;

}

return head.data;

}

4) Бінарний пошук за правилом золотого перерізу

//бінарний пошук удосконалений

int left = 0, right = size - 1, middle = 0;

bool found = false;

while (left < right && !found)

{

middle = Convert.ToInt32(right - (right - left) / 1.618);

if (arr[middle] == key) found = true;

else if (arr[middle] > key) right = middle - 1;

else left = middle + 1;

}

if (!found)

{

Console.WriteLine("Element not exist!");

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

else

{

Console.WriteLine("Element exist! Index = {0}", middle);

Console.WriteLine("Duration = " + sWatch.ElapsedMilliseconds + " ms");

}

Допоміжний метод для сортування списку

static void Sort(Node clas)

{

//сортування списку

int tmp;

Node index = null;

while (clas != null)

{

index = clas.next;

while (index != null)

{

if (clas.data > index.data)

{

tmp = clas.data;

clas.data = index.data;

index.data = tmp;

}

index = index.next;

}

clas = clas.next;

}

}

1. **Текст програми**

https://github.com/Virtuoso279/Maiboroda\_Igor.git

1. **Набір тестів**
2. Лінійний пошук (масив)

Кількість елементів: 20

Елемент для пошуку: -9

Індекс (елемент знайдено): 4

Масив: -439 404 286 186 -9 202 -456 428 83 191 153 387 -442 232 -273 -72 248 -201 -57 -241

1. Лінійний пошук з бар’єром (масив)

Кількість елементів: 50

Елемент для пошуку: -1

Індекс (елемент знайдено): 18

Масив: -49 355 224 470 -366 439 147 95 -315 228 -438 -358 234 139 -136 -344 -382 409 -1 -224 -244 20 376 -57 -281 -497 310 147 488 -78 -267 -163 -60 -497 414 -407 174 130 -112 130 307 -490 277 155 382 -393 -256 499 102 56

1. Бінарний пошук (масив)

Кількість елементів: 100

Елемент для пошуку: 81

Індекс (елемент знайдено): 58

Масив (відсортований): -489 -487 -482 -472 -463 -457 -445 -444 -438 -398 -378 -377 -355 -346 -336 -329 -326 -325 -323 -320 -269 -221 -218 -216 -207 -206 -189 -160 -149 -144 -126 -119 -116 -89 -82 -73 -70 -66 -52 -50 -45 -44 -42 -37 -33 -31 -3 7 9 11 20 22 29 34 59 61 64 78 81 92 99 99 104 109 120 121 124 144 151 153 174 176 181 189 201 207 207 212 217 230 238 246 272 278 298 304 305 311 312 349 362 370 382 395 405 408 409 445 475 487

1. Бінарний пошук за правилом золотого перерізу (масив)

Кількість елементів: 1000

Елемент для пошуку: 291

Індекс (елемент знайдено): 779

Масив (відсортований): -500 -499 -498 -498 -498 -498 -497 -497 -497 -497 -496 -495 -495 -494 -494 -494 -493 -492 -492 -491 -487 -486 -486 -486 -485 -482 -480 -479 -478 -478 -478 -477 -476 -475 -475 -473 -473 -471 -469 -467 -463 -462 -461 -457 -457 -456 -455 -453 -450 -450 -449 -449 -447 -447 -447 -447 -447 -445 -445 -445 -443 -442 -440 -438 -436 -435 -435 -434 -433 -433 -432 -431 -431 -430 -429 -429 -427 -426 -426 -425 -425 -424 -422 -421 -421 -420 -417 -417 -417 -413 -411 -411 -410 -408 -406 -404 -403 -403 -403 -402 -400 -397 -397 -394 -392 -390 -389 -388 -386 -385 -383 -382 -381 -380 -379 -379 -378 -377 -376 -375 -373 -373 -372 -372 -371 -370 -370 -369 -368 -367 -367 -366 -364 -363 -362 -360 -360 -358 -357 -357 -355 -355 -354 -354 -353 -353 -353 -352 -351 -350 -350 -350 -347 -347 -346 -344 -341 -340 -333 -333 -333 -332 -332 -331 -329 -328 -327 -326 -325 -325 -325 -325 -323 -323 -323 -322 -321 -320 -320 -319 -317 -317 -316 -315 -314 -312 -312 -312 -312 -311 -310 -310 -309 -308 -307 -307 -305 -302 -301 -298 -296 -295 -294 -293 -290 -290 -289 -289 -288 -287 -285 -285 -284 -284 -284 -284 -283 -283 -281 -281 -279 -278 -278 -278 -276 -276 -276 -275 -274 -274 -273 -272 -272 -271 -271 -271 -270 -269 -269 -269 -267 -266 -266 -265 -265 -265 -265 -265 -264 -261 -259 -257 -257 -256 -255 -252 -252 -252 -250 -250 -250 -249 -249 -249 -246 -244 -244 -242 -240 -238 -237 -236 -234 -234 -233 -232 -231 -229 -227 -227 -227 -227 -224 -223 -223 -223 -223 -222 -222 -222 -222 -221 -218 -217 -216 -215 -214 -213 -213 -213 -208 -206 -205 -205 -204 -204 -203 -202 -202 -201 -201 -199 -199 -199 -197 -196 -195 -195 -193 -191 -191 -190 -189 -188 -187 -186 -184 -184 -181 -181 -180 -179 -177 -176 -176 -175 -174 -173 -171 -171 -170 -170 -170 -169 -169 -169 -168 -167 -167 -166 -165 -164 -164 -163 -163 -162 -162 -161 -161 -160 -160 -159 -159 -157 -156 -155 -154 -154 -152 -152 -150 -149 -148 -148 -147 -147 -143 -140 -140 -140 -137 -137 -136 -135 -135 -135 -135 -133 -132 -131 -131 -129 -128 -125 -124 -124 -121 -120 -118 -117 -116 -115 -114 -113 -105 -104 -104 -102 -102 -98 -98 -98 -95 -95 -95 -95 -94 -93 -92 -91 -90 -90 -89 -88 -86 -85 -84 -83 -82 -80 -80 -79 -78 -77 -76 -76 -73 -73 -72 -71 -70 -70 -70 -68 -67 -67 -66 -66 -66 -64 -63 -61 -58 -57 -56 -56 -55 -55 -50 -50 -50 -49 -49 -47 -47 -44 -43 -42 -40 -39 -38 -38 -36 -35 -35 -35 -30 -29 -29 -28 -28 -27 -25 -24 -22 -22 -20 -20 -19 -15 -15 -14 -11 -10 -10 -9 -8 -8 -7 -6 -4 -3 -2 3 5 6 6 6 8 11 12 12 13 13 14 14 15 17 17 18 18 20 20 21 23 23 24 30 31 31 32 32 34 35 37 37 38 40 41 42 43 43 43 45 45 46 46 47 49 49 49 50 51 53 54 55 55 55 56 58 61 62 62 63 63 64 64 65 65 66 66 68 68 68 72 73 74 75 75 77 78 81 82 85 85 87 87 88 90 91 91 91 92 93 94 94 95 96 99 100 100 101 101 102 102 102 103 103 103 104 105 105 106 107 107 108 108 108 109 109 110 110 112 115 117 117 119 120 122 124 125 126 128 128 129 131 131 132 132 135 139 139 140 140 142 144 144 146 147 149 149 150 150 151 155 156 157 158 160 160 163 165 165 166 168 168 169 172 172 172 173 179 180 181 181 182 182 183 184 184 184 184 187 188 189 189 189 190 191 191 191 193 193 193 194 195 198 198 199 203 206 208 209 209 213 213 214 216 217 217 217 219 220 221 223 224 227 227 227 229 229 230 231 232 232 234 235 236 236 237 238 241 243 243 243 243 245 245 245 246 249 250 251 253 254 255 256 258 258 261 261 261 266 266 267 267 269 269 271 271 271 272 274 275 277 277 277 279 283 284 285 285 285 287 287 289 289 290 290 291 291 292 293 295 297 297 298 298 299 299 300 302 303 303 304 305 306 307 307 308 309 311 312 312 314 315 315 315 315 316 316 316 316 317 319 320 320 324 325 325 325 326 327 327 329 331 331 332 335 338 339 339 342 344 345 346 347 347 347 347 348 348 353 353 354 354 355 356 356 356 358 359 359 361 361 362 364 365 365 365 366 366 366 367 367 368 368 370 372 372 372 373 376 378 378 380 380 382 382 382 383 385 386 387 389 390 391 393 394 396 396 396 398 399 400 401 401 402 402 404 405 406 406 408 408 409 410 411 411 411 411 413 413 414 415 415 416 416 417 417 417 418 419 419 420 421 421 421 422 423 424 424 425 427 429 430 434 434 435 436 437 438 441 442 446 449 450 451 452 452 454 454 454 455 455 456 457 457 458 458 459 459 460 462 463 463 464 465 466 466 467 467 467 468 470 470 470 472 472 475 475 476 477 477 477 481 482 482 484 484 484 485 490 490 491 493 493 493 497 499

1. Лінійний пошук (linked list)

Кількість елементів: 20

Елемент для пошуку: -5

Індекс (елемент НЕ знайдено):

Linked list: -744 -372 144 -77 -179 92 -201 -53 64 177 443 -351 -307 -162 118 -351 -380 456 414 -331

1. Лінійний пошук з бар’єром (linked list)

Кількість елементів: 50

Елемент для пошуку: 337

Індекс (елемент знайдено): 44

Linked list: -764 -382 165 -88 96 107 -37 29 -201 398 -183 -230 388 148 191 -438 499 193 -29 285 -448 41 -458 135 437 435 52 419 124 42 336 -147 -25 -420 -57 109 -27 397 111 255 32 -480 -83 -296 337 385 -114 -278 163 -140

1. Бінарний пошук (linked list)

Кількість елементів: 100

Елемент для пошуку: 1

Індекс (елемент знайдено): 51

Linked list: -458 -449 -440 -410 -410 -402 -399 -398 -390 -382 -374 -356 -348 -329 -313 -311 -309 -293 -291 -288 -237 -218 -213 -211 -192 -189 -184 -176 -173 -170 -170 -159 -156 -153 -143 -135 -131 -129 -123 -123 -119 -99 -66 -58 -57 -53 -52 -27 -21 -21 -6 1 15 16 17 39 40 49 81 88 99 100 101 107 119 155 187 201 207 207 245 260 263 269 278 302 309 312 322 331 333 340 345 353 360 361 380 393 394 397 406 422 427 435 446 458 466 466 470 480

1. Бінарний пошук за правилом золотого перерізу (linked list)

Кількість елементів: 1000

Елемент для пошуку: 8

Індекс (елемент НЕ знайдено):

Linked list: -500 -500 -499 -497 -495 -494 -494 -493 -493 -492 -490 -489 -488 -488 -486 -484 -483 -483 -481 -481 -481 -479 -479 -478 -475 -474 -473 -473 -470 -469 -467 -465 -465 -462 -462 -461 -461 -461 -459 -458 -458 -457 -455 -451 -451 -451 -449 -446 -445 -442 -441 -440 -439 -438 -438 -437 -437 -436 -435 -433 -432 -432 -431 -431 -430 -427 -425 -425 -424 -423 -422 -421 -421 -420 -419 -418 -416 -415 -414 -414 -414 -413 -412 -412 -411 -411 -410 -410 -410 -410 -410 -409 -408 -408 -407 -406 -406 -406 -404 -404 -404 -403 -403 -403 -402 -401 -401 -400 -399 -399 -398 -397 -397 -396 -396 -396 -395 -395 -394 -393 -391 -391 -391 -390 -388 -387 -386 -385 -384 -384 -382 -382 -379 -374 -374 -372 -372 -371 -370 -370 -368 -366 -363 -363 -363 -362 -360 -360 -359 -358 -358 -354 -351 -349 -348 -344 -343 -343 -341 -341 -340 -340 -340 -339 -338 -338 -337 -337 -337 -336 -336 -334 -332 -326 -326 -325 -322 -317 -317 -317 -316 -316 -312 -312 -312 -310 -307 -306 -305 -304 -304 -303 -302 -301 -300 -299 -298 -297 -297 -296 -296 -293 -291 -290 -290 -289 -289 -286 -286 -285 -284 -279 -278 -278 -277 -275 -275 -275 -275 -274 -273 -273 -273 -271 -270 -267 -266 -265 -265 -265 -264 -263 -263 -263 -262 -262 -258 -255 -254 -254 -254 -252 -250 -250 -249 -248 -247 -247 -245 -243 -243 -238 -235 -235 -234 -231 -231 -230 -230 -229 -228 -228 -227 -227 -226 -223 -223 -222 -222 -222 -221 -220 -217 -216 -216 -216 -216 -216 -216 -216 -214 -212 -206 -206 -205 -205 -204 -202 -200 -200 -197 -197 -196 -196 -196 -196 -195 -194 -191 -191 -191 -187 -187 -185 -184 -184 -184 -184 -182 -180 -178 -178 -177 -175 -170 -169 -169 -169 -169 -169 -167 -167 -166 -165 -163 -163 -162 -162 -161 -160 -160 -159 -157 -157 -156 -154 -154 -153 -151 -150 -149 -148 -145 -144 -143 -142 -141 -139 -139 -138 -138 -137 -136 -135 -133 -132 -131 -131 -128 -128 -124 -123 -123 -122 -119 -118 -114 -114 -114 -112 -109 -108 -108 -107 -107 -104 -104 -104 -103 -102 -102 -99 -99 -98 -98 -98 -97 -96 -96 -95 -95 -94 -94 -93 -92 -90 -89 -87 -87 -87 -85 -85 -83 -82 -80 -78 -75 -75 -74 -74 -73 -72 -72 -72 -71 -70 -70 -67 -67 -67 -64 -63 -61 -60 -59 -59 -57 -55 -53 -53 -51 -49 -49 -48 -46 -45 -45 -43 -42 -42 -42 -41 -40 -40 -39 -39 -38 -36 -35 -34 -33 -32 -30 -30 -25 -19 -19 -19 -19 -18 -14 -12 -12 -10 -10 -10 -9 -9 -9 -8 -8 -7 -6 -2 -1 -1 -1 -1 0 1 2 2 2 3 4 5 6 6 6 7 7 7 9 9 10 11 11 15 15 15 17 18 20 20 22 22 23 23 23 24 25 25 26 26 28 29 30 30 30 31 32 32 33 34 34 35 35 35 36 36 37 38 38 38 39 41 41 41 41 42 42 45 45 46 47 48 51 52 52 55 55 55 56 56 56 57 58 59 60 61 62 63 64 65 67 69 71 71 72 73 73 75 76 77 78 78 80 80 81 81 81 81 83 84 86 86 87 89 92 93 94 97 97 99 99 102 102 104 105 105 107 109 111 111 114 115 118 118 118 119 120 121 121 121 121 122 122 125 126 127 127 127 128 128 133 135 135 136 136 136 137 138 138 138 139 139 141 143 145 145 146 147 147 148 152 152 153 153 153 154 154 156 158 158 159 161 161 162 164 164 164 165 165 166 167 168 169 170 171 173 173 174 176 179 184 185 185 185 187 187 191 192 193 197 198 199 200 200 203 207 207 207 207 210 210 211 212 213 215 215 219 219 220 222 222 223 225 226 227 227 228 228 228 228 231 231 232 233 234 234 234 236 237 238 238 241 244 244 244 245 246 247 248 249 252 253 253 255 255 256 257 257 259 259 261 261 262 263 264 265 265 267 268 268 268 270 271 271 271 271 271 272 272 273 274 274 276 276 276 276 277 279 280 280 282 282 284 285 286 288 289 291 292 295 296 296 297 297 298 298 300 301 302 303 304 306 307 307 308 311 311 312 313 313 315 316 316 317 320 320 320 321 323 323 326 328 328 329 330 331 333 333 336 337 338 340 341 341 344 344 345 346 346 347 347 347 348 348 349 349 350 350 350 351 351 357 360 360 360 361 361 361 365 365 366 366 368 369 370 371 371 372 374 374 375 375 376 376 376 378 379 380 385 387 387 388 389 391 392 393 395 397 397 397 399 400 400 400 402 403 404 404 405 405 406 406 407 409 411 411 413 414 415 418 418 419 419 419 420 420 420 421 422 422 423 424 424 425 425 427 428 429 429 430 430 430 430 431 436 436 436 437 438 438 440 440 441 441 442 442 443 444 445 445 447 448 450 451 452 452 453 453 454 455 459 460 460 461 461 461 461 462 463 464 464 467 467 467 468 468 468 469 471 471 472 475 476 478 478 479 482 484 485 485 486 487 488 488 489 490 491 491 491 492 493 494 497 497 498 499

1. **Результати тестування і аналіз**
2. Лінійний пошук

|  |  |
| --- | --- |
| Масив: | Лінійний пошук: |

Отже, ефективність лінійного пошуку більша саме в масиві, ніж в linked list

1. Лінійний пошук з бар’єром

|  |  |
| --- | --- |
| Масив: | Лінійний список: |

Отже, лінійний пошук з бар’єром працює в 2 рази швидше в linked list, ніж в масиві.

1. Бінарний пошук

|  |  |
| --- | --- |
| Масив: | Лінійний список: |

Отже бінарний пошук в масиві працює швидше, ніж в лінійному списку

1. Бінарний пошук за правилом золотого перерізу

|  |  |
| --- | --- |
| Масив: | Лінійний список: |

Отже, за правилом золотого перерізу бінарний пошук швидше працює саме в лінійному списку.

1. **Висновок**

Якщо подивитися на всі результати, то бінарний пошук працює довше, ніж лінійний через те, що ми маємо додатково сортувати масив чи список и виводити значення на консоль ще раз. Якщо взяти сам алгоритм, то він буде працювати набагато швидше, ніж алгоритм лінійного пошуку, час виконання алгоритму бінарного пошуку буде менше 1 мілісекунди, а лінійного приблизно 1-2 мілісекунди при великої кількості чисел.