21AIE111

Data Structures & Algorithms

M.Kalyana Sundaram

CB.EN.U4AIE21120

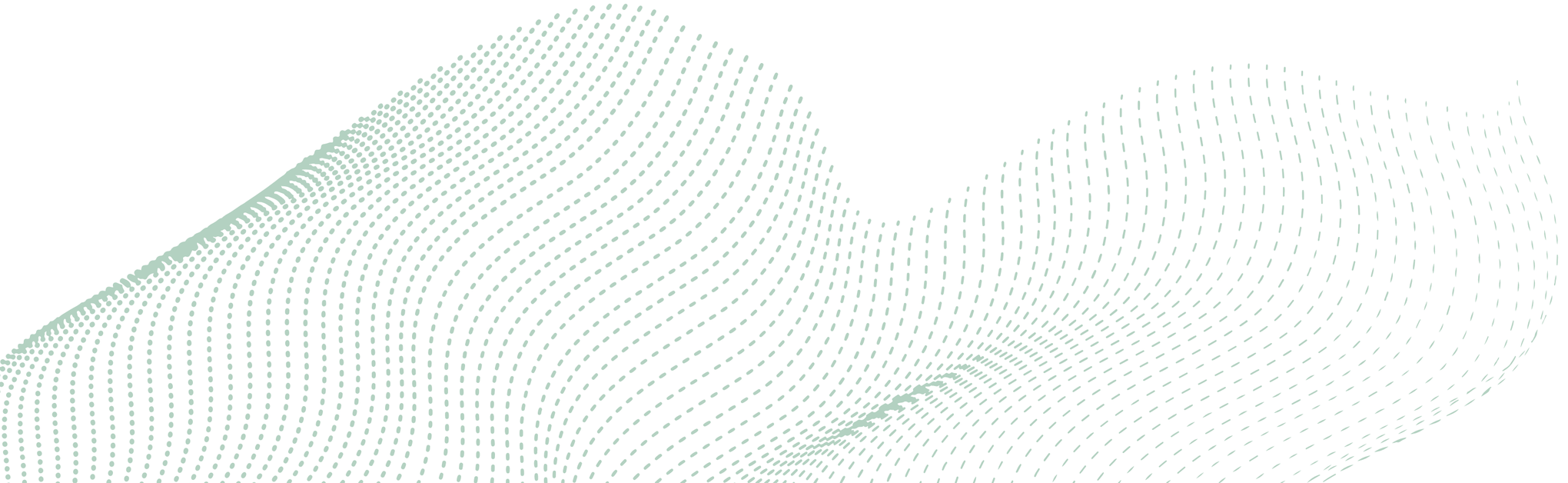
1. Write a java code to generate 1 lakh random integer numbers between 1 and 1 lakh.

   a) Collect it to an array.

   b) Pick a number in the far end of the array

   c) search that number and observe the time taken (in seconds)

   (you can increase the total numbers to 10 lakh also)



Code & Output

Snippet

```java
public class q1 {
        public static void main(String[] args) {
            //making array and assigning random numbers
            int[] arr = new int[1000000];
            for (int i = 0; i < arr.length; i++) {
                arr[i] = (int) (Math.random() * 1000000);
            }
            int arrlen = arr.length;
            int ln = arr[arrlen-1];
            System.out.println("Last element is "+ln);
            //time taken to search the last element
            long startTime = System.currentTimeMillis();
            for (int i = 0; i < arrlen; i++) {
                if (arr[i] == ln) {
                    System.out.println("Last element found at index "+i);
                    break;
                }
            }
            long endTime = System.currentTimeMillis();
            long timeElapsed = endTime - startTime;
            System.out.println("Time taken to search the last element is "+timeElapsed+"
milliseconds");
        }
    }
```

```
~/Documents/Assignment4                                  base  at 04:02:08 PM
java q1.java
Last element is 773915
Last element found at index 2927
Time taken to search the last element is 0 milliseconds

~/Documents/Assignment4                                  base  at 04:02:17 PM
neofetch
                    `.:/ossyyyysso/:.
                .:oyyyyyyyyyyyyyyyyyo:`
             -oyyyyyyyodMMyyyyyyysyyyyo-
          -syyyyyyyyyydMMyoyyyydmMMyyyyys-
       oyyysdMysyyyydMMMMMMMMMMMyyyyyyyo
     `oyyyydMMMMysyysooooooodMMMMyyyyyyyyo`
    oyyyyyydMMMMyyyyyyyyyyyysdMMysssssyyyo
   -yyyyyyyydMysyyyyyyyyyyyyyysdMMMMysyyy-
   oyyyysoodMyyyyyyyyyyyyyyyyyyydMMMMysyyyo
   yyysdMMMMMyyyyyyyyyyyyyyyyyyysosyyyyyyyy
   yyysdMMMMMyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy
   oyyyyysosdyyyyyyyyyyyyyyyyyydMMMMysyyyo
   -yyyyyyyydMysyyyyyyyyyyyyyysdMMMMysyyy-
    oyyyyyydMMMysyyyyyyyyyyysdMMyoyyyoyyyo
     `oyyyydMMMysyyyooooooodMMMMyoyyyyyyyo
       oyyysyyoyyyysdMMMMMMMMMMyyyyyyyyo
         -syyyyyyyyydMMMysyyydMMMysyyyys-
            -oyyyyyyydMMMyyyyyysosyyyyo-
               ./oyyyyyyyyyyyyyyyyyo/.
                  `.:/oosyyyysso/:.`
                                        naylak15@naylaK
                                        ---------------
                                        OS: Kubuntu 21.10 x86_64
                                        Host: OMEN by HP Laptop 15-dc1xxx
                                        Kernel: 5.13.0-30-generic
                                        Uptime: 22 mins
                                        Packages: 2427 (dpkg)
                                        Shell: zsh 5.8
                                        Resolution: 1920x1080
                                        DE: Plasma 5.22.5
                                        WM: KWin
                                        WM Theme: Sweet-Dark
                                        Theme: Sweet [Plasma], Breeze [GTK3]
                                        Icons: [Plasma], candy-icons [GTK2/3]
                                        Terminal: dolphin
                                        CPU: Intel i7-9750H (12) @ 4.500GHz
                                        GPU: NVIDIA GeForce GTX 1650 Mobile / Max-Q
                                        GPU: Intel CoffeeLake-H GT2 [UHD Graphics 630]
                                        Memory: 5470MiB / 7755MiB
```

Output and System Config

2. Write a java code to generate 1 lakh random integer numbers between 1 and 1 lakh.

   a) Collect it to a single linked list.

   b) Pick a number in the far end of the list (traverse the list)

   c) search that number and observe the time taken (in seconds)

(you can increase the total numbers to 10 lakh also)

Code

Snippet

```java
import java.sql.Time;

public class q2 {
    Node head;
    public static class Node {
        int data;
        Node next;
        Node(int d)
        {
            data = d;
            next = null;
        }
    }
    // Function to insert a new node at the beginning
    public void push(int new_data)
    {
        Node new_node = new Node(new_data);
        new_node.next = head;
        head = new_node;
    }
    // Function to print nodes in a given linked list
    public void printList()
    {
        Node tnode = head;
        while (tnode != null)
        {
            System.out.print(tnode.data+" ");
            tnode = tnode.next;
        }
    }
```

```java
    //function to assign random numbers to nodes from 1 to 1 lakh
    public void assignRandom()
    {
        Node tnode = head;
        while (tnode != null)
        {
            tnode.data = (int)(Math.random()*1000000);
            tnode = tnode.next;
        }
    }
    public static void main(String[] args) {
        q2 llist = new q2();
        for (int i = 0; i < 100000; i++) {
            llist.push(i);
        }
        llist.assignRandom();
        llist.printList();
        //finding the last element of the list
        long startTime = System.currentTimeMillis();

        Node tnode = llist.head;
        while (tnode.next != null)
        {
            tnode = tnode.next;
        }
        long endTime = System.currentTimeMillis();
        long timeElapsed = endTime - startTime;
        System.out.println("\nThe last element of the list is: "+tnode.data);
        System.out.println("Time taken to find the last element of the list is:
"+timeElapsed+" milliseconds");
    }

}
```

Output

```
4 292612 315103 646409 220617 310313 330514 646297 412607 126176 514269 864792 926068 965370 416287 353815 852432 60762 900523 351897 745359 873629 534329 628097 358280 315188 155921 466533 589098 74935 76763 710481 202651 670558 298546
 684446 340000 374255 328512 743261 43314 741549 662276 659166 26149 71248 769841 51025 701085 758988 468397 584194 40211 171825 894586 725286 188645 367633 899236 422922 877265 710621 579232 203793 588051 193046 833966 822695 113911 63
259 287982 785947 624598 181490 656732 779427 601298 377343 648982 275715 34685 966733 797265 369165 94744 501058 186859 193453 75346 767677 914135 113023 792591 38303 246726 248043 773098 570719 880795 791877 506961 409253 394811 10757
 759765 600744 984579 777242 439064 45263 830863 982237 792936 334659 998813 194719 781714 97009 597496 494024 308705 561378 38180 486439 621710 284792 621628 136011 604357 242161 357514 188897 864747 769886 873161 651257 498659 175206
504167 992581 155492 175858 176075 489539 567399 718419 484109 399812 15505 138755 457076 461076 301846 74248 851919 547191 671648 15067 466966 181810 787771 890496 884818 9071 297639 626374 259880 251657 752697 181028 250908 648446 641
409 169573 965562 357144 79034 820261 831765 831676 260922 671615 639821 739820 130743 115351 609549 573441 841109 617628 228328 721864 491202 209991 778780 428351 314571 631355 928917 230285 959916 323717 492949 195594 758716 336801 48
7535 511124 905739 80623 906105 753042 291975 780009 213010 475071 324252 431410 419579 587223 441762 39847 745777 365889 196968 922659 294012 835556 706829 906739 865736 693064 3 317234 36830 907265 286677 758232 874386 620079 459886 8
19738 225323 999370 200673 10302 336395 249729 904153 410906 675496 74165 269536 773319 215763 497864 804383 44856 116835 114592 681841 591562 691731 496990 359341 713104 783149 723909 118207 395204 189603 721114 215867 629240 579653 38
6073 655811 43660 411615 273254 675190 38177 736428 953221 925905 89215 403038 27010 982297 403887 108435 176684 246966 639834 938638 563606 546043 979508 532230 347913 706160 450059 764680 95810 187624 13755 899531 365743 411759 618901
 481608 820143 765106 97526 857808 796392 389178 523634 102571 388768 771634 404607 298037 742600 843282 868393 572952 442482 754995 830390 39780 588324 759621 976490 786039 396210 656197 282514 373755 935381 67249 705346 880026 918406
414690 627315 372037 298455 835786 612952 851462 795771 437929 46019 126058 374880 229375 254755 770336 124687 782894 804560 322561 955582 190193 739894 131937 20785 147254 690946 817657 793619 74170 925671 130651 288044 17806 624072 11
9151 996691 711576 879175 890854 738827 573872 187880 937282 760057 615131 484967 940041 106271 193285 94789 214049 506241 308286 327725 459380 740420 658638 832286 291261 747952 996350 228756 96806 529521 411439 996591 127481 240610 53
095 65292 377847 737890 451843 67300 724014 240890 45519 751459 999318 701738 731602 626917 170098 937697 269407 509843 541044 574384 137902 279169 967448 371960 18350 812542 380771 562249 415733 770031 544322 411676 58881 620624 847418
 343039 626452 65345 119253 598588 34389 800894 897830 607521 990207 388951 324606 932915 770449 848869 852181 916310 209850 475240 250121 912535 704251 296876 973195 313057 624027 540024 816593 393038 910754 436177 924325 385454 787081
 723637 639850 85036 448251 232018 392066 715463 323429 10169 795181 755157 995980 560532 407539 910170 154073 876750 142371 447111 6444 146370 152514 654668 442899 176769 121502 289251 845101 829993 285032 478393 860417 10955 179673 70
5020 640072 43517 69761 176292 12601 590220 238798 92705 814529 624025 848271 76252 132491 344160 864637 290886 524418 343244 634362 594182 382200 779139 47211 396007 474169 782502 869613 768170 362996 839842 752773 748459 833062 431863
 624301 713096 681103 778224 657471 153593 224882 142354 778853 187137 666891 783740 722872 654053 13549 310186 970984 711546 595097 878778 523500 419860 357352 421875 343674 935127 3741 833198 486549 572626 754461 839168 379145 184800
959398 101878 453913 10326 52364 952689 353999 393675 366466 39173 226708 549023 143892 83196 200654 848033 668108 894209 2431 947952 822480 759370 810261 444615 536501 340889 864906 634376 519333 251037 62235 399643 400072 546552 73719
9 194615 895132 486719 115569 600963 905356 33157 457620 947654 508964 734443 747085 44718 661147 834928 889010 167741 37109 674693 771780 47969 862589 706345 221428 772375 764350 744415 830409 178657 642873 384397 878433 542095 320462
880999 79488 94318 250182 159004 687088 188474 884540 45116 568847 730956 397556 255512 221853 836725 329808 151339 341906 533485 383322 783160 482533 739324 31584 285445 543820 669139 829750 253439 316411 287060 311042 337307 72661 415
256 137590 922244 244761 633914 737438 428271 558776 162856 4150 459717 603853 502172 761453 761951 707498 727395 602559 701698 160643 984186 402296 777314 290345 908130 269892 486689 613285 407184 961254 732988 319882 487472 801790 832
224 636626 577527 984309 101689 193394 111761 701253 954579 382837 970844 946508 403670 840516 744847 979243 885693 597348 684415 380724 358106 160000 635892 354767 691236 760226 218248 423470 481315 280754 992809 952091 543713 913164 4
01221 424 510723 432318 802722 842738 154519 82236 28723 124324 283485 459933 490875 29937 318848 456567 28176 27873 948656 269359 359384 723163 577356 733090 218868 104338 641831 736598 997240 443 248500 287523 982481 878469 1150 86767
7 102974 206065 335442 81745 410742 480418 153584 243647 411385 681363 61439 467713 746582 345303 50398 35960 665738 676224 37608 298080 185223 111070 280708 127370 826843 538977 316997 3675 351598 937325 742025 123525 960415 532321 838
838 547993 20506 717770 796708 290400 494122 872943 16915 418858 895909 908918 793220 702725 785736 842489 475941 67411 515777 947735 81095 138573 193779 67542 189843 454582 598894 480868 898531 505256 92081 316260 525885 615199 400022
878108 761503 831688 519655 409338 948503 399898 889832 752756 233312 112373 407288 92934 386788 804977 826011 461542 203470 538175 572265 971243 519502 550363 31951 806746 54829 899016 922623 846340 481552 815843 449424 413197 347867 1
93675 562346 62985 745806 295714 781645 502780 901021 731833 738905 706011 184175 982008 793145 39135 395992 263518 824555 747984 561865 336453 582835 465180 886915 921119 772784 494763 968697 761325 396701 506131 842308 9112 741332 926
637 941403 529338 796494 760662 598700 926657 823179 782682 280819 582741 250867 52879 354852 758235 76912 486004 817854
The last element of the list is: 817854
Time taken to find the last element of the list is: 1 milliseconds

~/Documents/Assignment4 ........................................ ✓ base at 04:10:33 PM
```

3. Write a java code to generate 1 lakh random integer numbers between 1 and 1 lakh.

a) Collect it to an array.

b) Pick 3 number in the far end of the array and create a sub-array

c) search the sub-array and observe the time taken (in seconds)

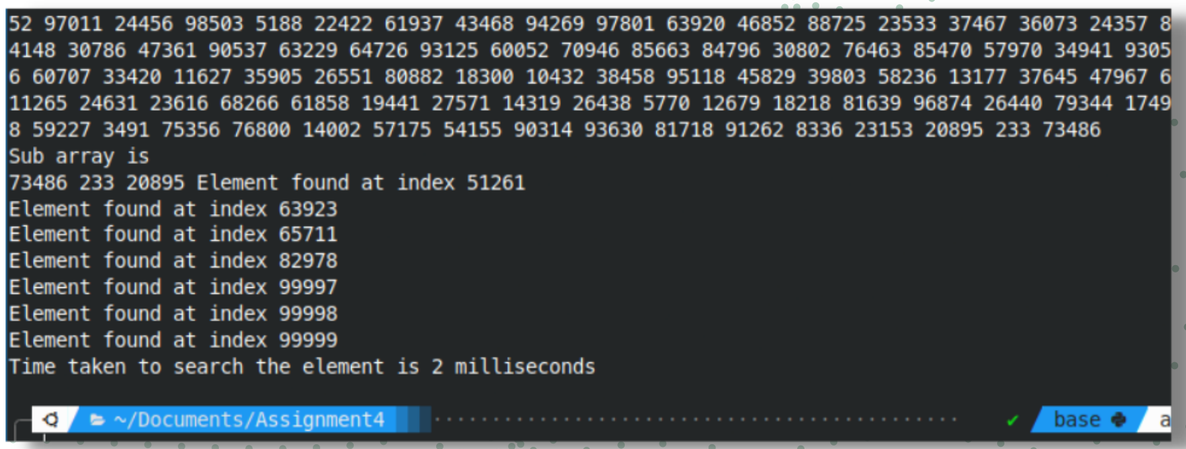
(you can increase the total numbers to 10 lakh also)

Code & Output

Snippet

```
public class q3 {
    public static void main(String[] args) {
        int[] arr = new int[100000];
        for (int i = 0; i < arr.length; i++) {
            arr[i] = (int) (Math.random() * 100000) + 1;
        }
        // print the array
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
        //pick 3 numbers from far end of the array
        int[] sub = new int[3];
        for (int i = 0; i < 3; i++) {
            sub[i] = arr[arr.length - 1 - i];
        }
        System.out.println("Sub array is ");
        // print the sub array
        for (int i = 0; i < sub.length; i++) {
            System.out.print(sub[i] + " ");
        }

        //search the element in the sub array from the main array
        long startTime = System.currentTimeMillis();
        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j < sub.length; j++) {
                if (arr[i] == sub[j]) {
                    System.out.println("Element found at index " + i);
                    break;
                }
            }
        }
        long endTime = System.currentTimeMillis();
        long timeElapsed = endTime - startTime;
        System.out.println("Time taken to search the element is "+timeElapsed+"
milliseconds");

    }


}
```

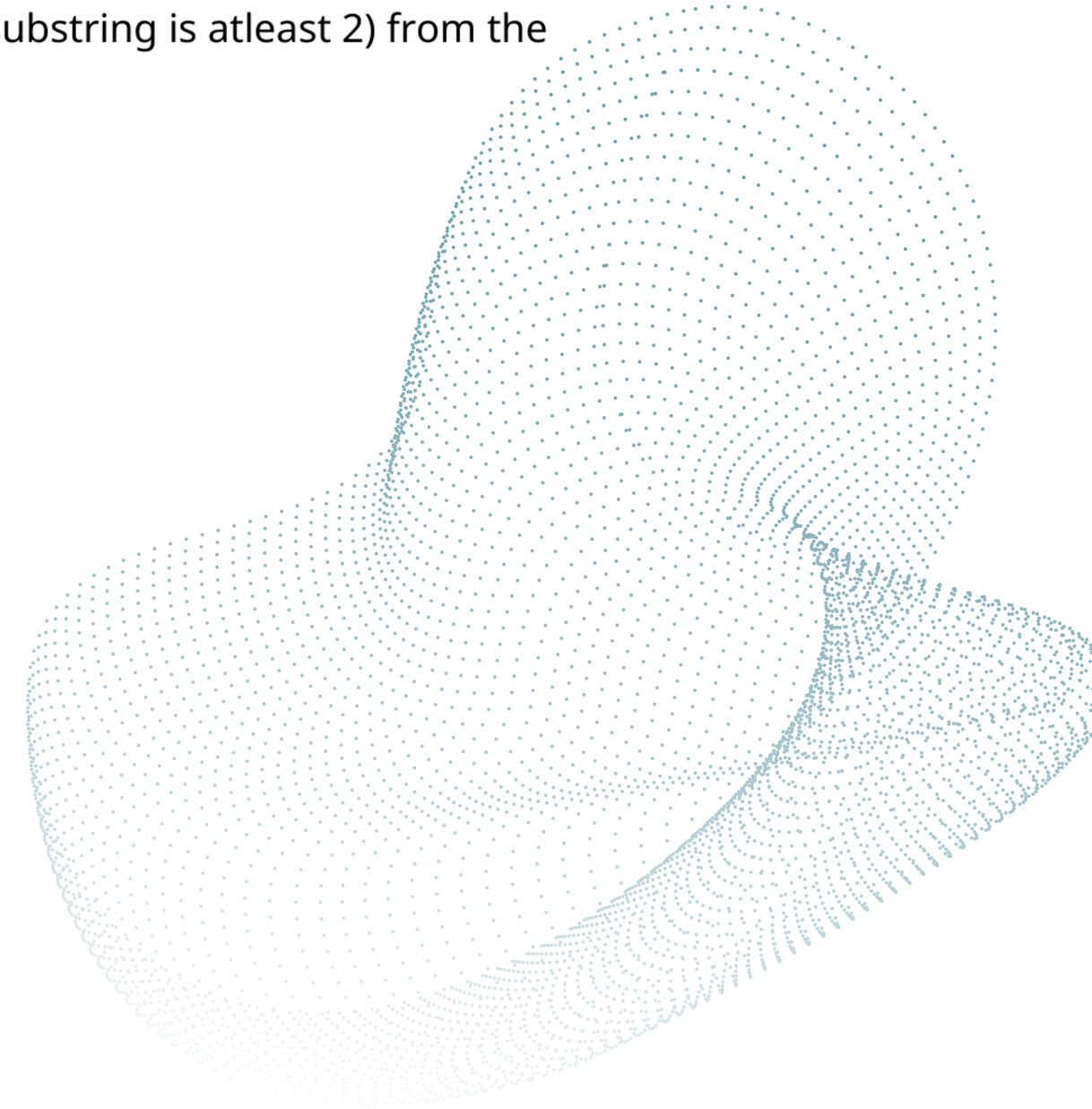
```
52 97011 24456 98503 5188 22422 61937 43468 94269 97801 63920 46852 88725 23533 37467 36073 24357 8
4148 30786 47361 90537 63229 64726 93125 60052 70946 85663 84796 30802 76463 85470 57970 34941 9305
6 60707 33420 11627 35905 26551 80882 18300 10432 38458 95118 45829 39803 58236 13177 37645 47967 6
11265 24631 23616 68266 61858 19441 27571 14319 26438 5770 12679 18218 81639 96874 26440 79344 1749
8 59227 3491 75356 76800 14002 57175 54155 90314 93630 81718 91262 8336 23153 20895 233 73486
Sub array is
73486 233 20895 Element found at index 51261
Element found at index 63923
Element found at index 65711
Element found at index 82978
Element found at index 99997
Element found at index 99998
Element found at index 99999
Time taken to search the element is 2 milliseconds
```



4. Write java code to find the possible substrings (size of substring is atleast 2) from the given sequences

a) ATGCT b) AGCT

a) CCGTCG b) CCGCG



Code & Output

Snippet

```java
import java.util.Scanner;

public class q4 {
        //making an array and inputing strings as elements
        public static void main(String[] args) {
            //getting n from user
            Scanner sc = new Scanner(System.in);
            System.out.println("Enter the number of elements in the array");
            int n = sc.nextInt();

            String[] arr = new String[n];
            //getting user input for strings and storing them in the array
            for (int i = 0; i < arr.length; i++) {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter a string: ");
                arr[i] = sc.nextLine();
            }
            //find possible substrings from given sequence
            Findsub(arr);

        }
        private static void Findsub(String[]arr) {
            //starting point of the substring
            System.out.println("Substring is ");
            for (int i = 0; i <= arr.length; i++) {
                //ending point of the substring
                for (int j = i; j <= arr.length; j++) {
                    //printing the substring of minimum size 2
                    if (j - i >= 2) {

                        for (int k = i; k < j; k++) {
                            System.out.print(arr[k] + " ");
                        }
                        System.out.print("");
                    }
                    System.out.println();
                }

            }

        }
    }
```

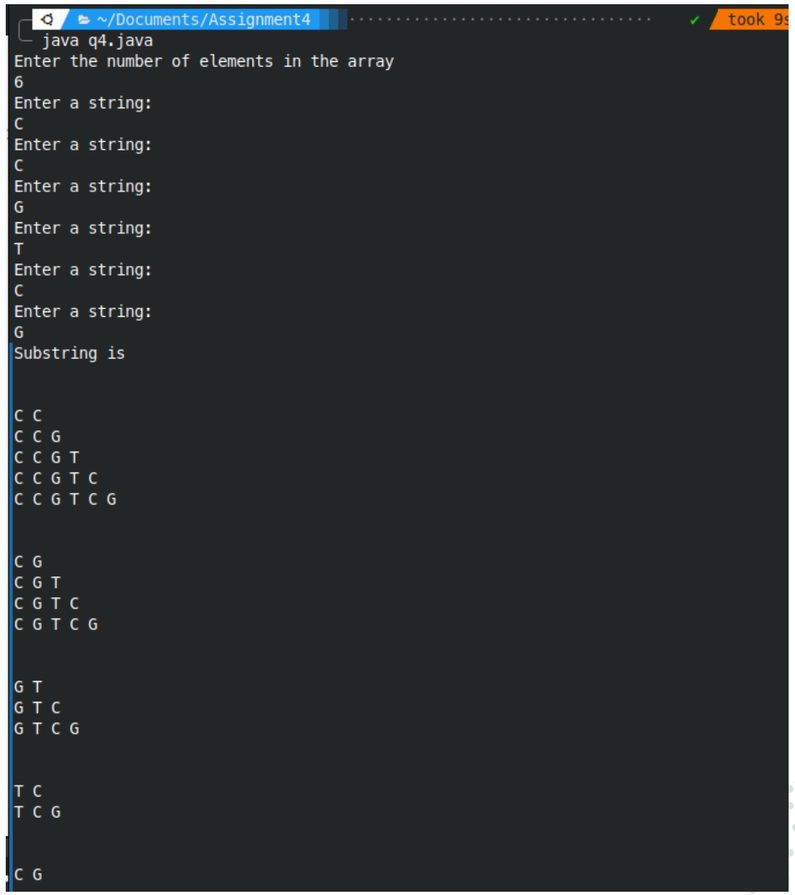
```
~/Documents/Assignment4 ..... INT ✘ took 5s base at 04:33:08
java q4.java
Enter the number of elements in the array
5
Enter a string:
A
Enter a string:
T
Enter a string:
G
Enter a string:
C
Enter a string:
T
Substring is


A T
A T G
A T G C
A T G C T


T G
T G C
T G C T


G C
G C T


C T
```

CASE 1

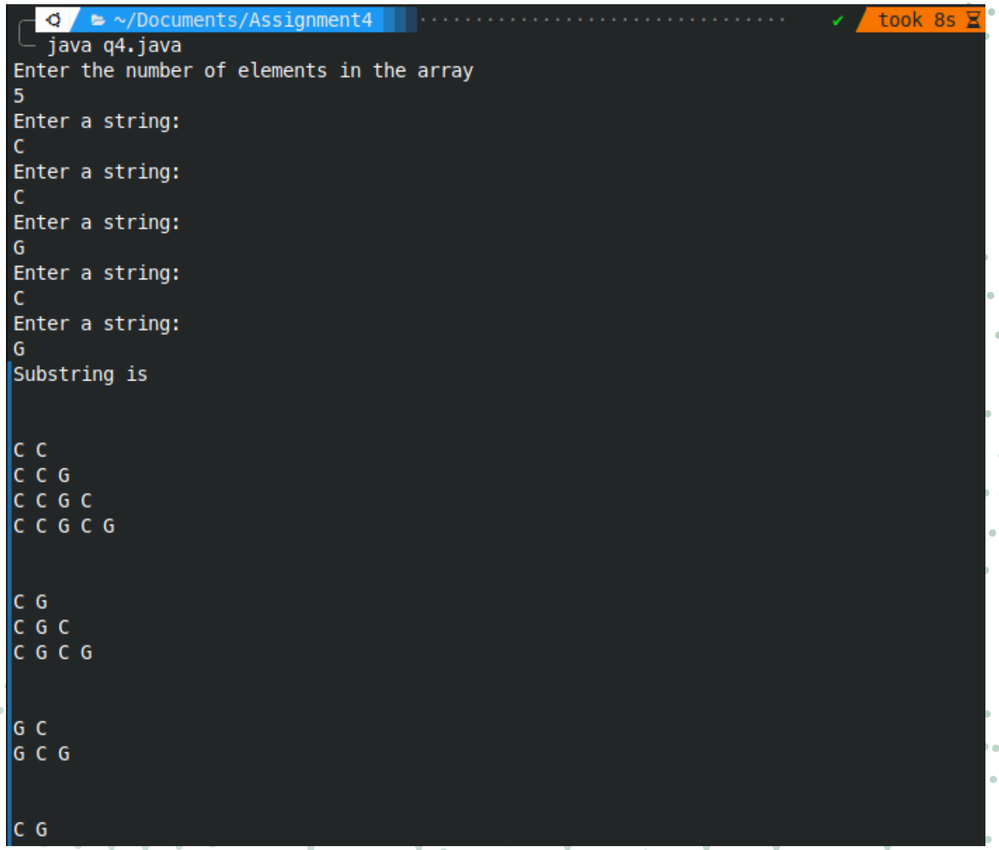
```
~/Documents/Assignment4 ..... ✓ took 5s
java q4.java
Enter the number of elements in the array
4
Enter a string:
A
Enter a string:
G
Enter a string:
C
Enter a string:
T
Substring is


A G
A G C
A G C T


G C
G C T


C T
```

CASE 2

OUTPUT



CASE 3



CASE 4

Thank you!