

**KOLEJ PROFESIONAL MARA BERANANG**

**DATA STRUCTURE & ALGORITHM**

**CSC2734**

**PREPARED FOR:**

PUAN SURIANA BINTI SUPAAN

**PREPARED BY:**

AISYAH AINA SUFIA

**MENTOR’S NAME:**

PUAN NOOR FAZLIANI BINTI SHAMSHUDIN

**TASK 1:**

Provide overview of your application including the objective of implementing the sorting and searching techniques for the application.

* OVERVIEW

Learning achievement is crucial as it is a representative of student understanding and success level in learning . The Students Score Application will support efficiency of data processing as it is designed suitable with among students and keep track the student’s development potential and organize report for teacher or lecturer to review . The features itself are consist to display scoresheets marks from highest to the lowest , update data, remove data and many more so that user feel more flexible with the menu options provided. It is more managerial for future strategic activities and provide to certain related field either inside or outside schools or organization with the necessary report.

**OBJECTIVE OF IMPLEMENTING**

* SORTING

Sorting is one of the most thoroughly studied algorithms in computer science. Sorting is arranging in an ordered sequence .There are dozens of different sorting implementations and applications that you can use to make code more efficient and effective.

By using sorting method, the sorting process is too easy and quickly locate an element in a sorted list and design an efficient algorithm. Also, this method impelement reorganizing a large number of items into specific order, such as alphabetical and highest-to-lowest.

* SEARCHING

Searching algorithm is an algorithm which solves a search problem.

Searching algorithm is designed to check an element or retrieve element from any data structure where it is stored. Other than that , it is also designed to do calculation about the search space of a problem domain either it is discrete or continuous vakyes

**TASK 2:**

* **SORTING : BubbleSort**

Definition: Bubble sort, also known as sinking sort, is a basic sorting algorithm that iterates through a list, comparing neighbouring entries and swapping them if they are out of order.

The reason BubbleSort was chosen because:

1. It is the most easiest method of sorting.to implement in application. Comparison-based algorithm in which each marks is compared and the next marks is swapped if they are not in order.
2. Sorting in place following order . When marks and name are sorted based on marks either lowest to highest or highest to lowest, its easier to read and detect highest and lowest marks.
3. Faster to understand eventhough not really used in real world. The reason I choose this because the appropriate search algorithm often depends on the data structure being searched which is the application commonly being used in classroom for one institute and it doesn’t required a lot of students. With the fewer code of line, to fix the problem is also easier to fix.

* **SEARCHING : SequentialSearch**

Definition : A method for locating an element within a list is called linear search or sequential search. It checks each element of the list one by one until a match is discovered or the entire list is searched.

The reason SequentialSearch was chosen because:

1. More easier to use . Comparison-based algorithm in which each marks is compared and the next marks is swapped if they are not in order. they still able to swap the marks or student name list eventhough it is not sorted accordingly. More suitable if related like a total student in class which not above than 40 students.
2. For me it is efficient with the condition requirement only involved with small data of student in classroom. Because there is given a list target value, the algorithm iterates through every entry on the list compares it to the target.

**TASK 5:**

1. **The Sorting Processes** – 5 students

[ Aisyah , Sufia , Aina , Adleen , Amirah ]

[ 96 , 61, 76 , 12 , 45]

**Scenario:** Sorted by the highest to the lowest

**(STEP 1):** Starts by comparing the first and second elements in the list. If the first element is smaller than second element, swap these two elements

But in this case, first element (in index 0) is greater than second element (which is index 1). Thus, no swap is occur.

0 1 2 3 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96  (1) | Sufia  61  (2) | Aina  76  (3) | Adleen  12  (4) | Amirah  45  (5) |

**(STEP 2):**  Now, start to compare the second element (61) with index 2 and third element(76) with index 3 in the list. The first element is smaller than the second element. So we need to swap these two element.

0 1 2 3 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96  (1) | Sufia  61  (2) | Aina  76  (3) | Adleen  12  (4) | Amirah  45  (5) |

**(STEP 3):**  Now, the item is already swap ( change to colour green ) . Lets compare the fourth element with third element. Third element which is number 61 , is greater than fourth element which is 12 . So we cant swap both of them.

0 1 2 3 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96  (1) | Aina  76  (2) | Sufia  61  (3) | Adleen  12  (4) | Amirah  45  (5) |

**(STEP 4):**  The list is starting to look sorted, but not complete yet . Now compare both the fourth element with fifth element. The fourth element which is in index 3 , is smaller than fifth element (in index 4) . We need to swap them. Or also we can say fifth element is larger than fourth element, so we can swap.

0 1 2 3 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96  (1) | Sufia  61  (2) | Aina  76  (3) | Adleen  12  (4) | Amirah  45  (5) |

**(STEP 5):**  Now the list has reached the end of of the list. It’s clearly that no more swaps need to be made because its already sorted and completed.

0 1 2 3 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96  (1) | Sufia  61  (2) | Aina  76  (3) | Adleen  12  (4) | Amirah  45  (5) |

* **CONCLUSION**

In a nutshell, BubbleSort will continue to swap elements until it compares every item in the list. The algorithm will not stop until every swap has taken the place.

**b.) The Searching Processes** – 5 students

[ Aisyah , Aina ,Sufia ,Amirah, Adleen ]

[ 96 , 76 , 61 , 45 , 12]

**Item search is: 45**

**Solution:**

**(STEP 1):** Starting from beginning which is index 0, check the first element in the list either it is number 45 or not.

**0 1 2 3 4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96 | Aina  76 | Sufia  61 | Amirah  45 | Adleen  12 |

**(STEP 2):** So the answer is no. So move to next element and check again whether it is 45 or not.

**0 1 2 3 4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96 | Aina  76 | Sufia  61 | Amirah  45 | Adleen  12 |

**(STEP 3):** Element number 76 is not 45 . Then, move to next element and compare it with the item search, which is 45.

**0 1 2 3 4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96 | Aina  76 | Sufia  61 | Amirah  45 | Adleen  12 |

**(STEP 4):** The answer is no. number 61 is not matching with number 45. Then we move to the next element and compare both of them again.

**0 1 2 3 4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96 | Aina  76 | Sufia  61 | Amirah  45 | Adleen  12 |

**(STEP 5):** Now compare the current number with search item number. Is the number 45 is same with the item that searching for? True. So the item is found at index 3.

**0 1 2 3 4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aisyah  96 | Aina  76 | Sufia  61 | Amirah  45 | Adleen  12 |

* **CONCLUSION:**

The idea of linear searching is simple, which is the item search number must go through each of elements to compare with in the list, in order, until we find the correct value.

**TASK 7 : REPORT**

1. Coding code

#user choose one of the task

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print(" ----- WELOCOME TO STUDENT SCORE APPLICATION----- - ")

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

#enter name of class & institution. Act as a header form

print("Information About The Institution")

sch = input("School/University Name:")

classes = input("Class: ")

year = int(input("Year of: "))

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("\u2764\uFE0F","MENU","\u2764\uFE0F")

print ("\n1.) ADD STUDENTS DATA \n2.) SEARCH A STUDENT AND UPDATE MARKS \n3.) DISPLAY HIGHEST AND LOWEST MARK \n4.) EXIT")

print ("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

choice = "yes"

while choice == "yes":

option = int(input("\nPlease select options menu above (number): "))

#user have to key in names and score

if option == 1:

print ("please enter student's name and mark")

liststudent = []

listsmark = []

#enter the number of students

n = 0

n = int(input("How many students?: "))

for i in range (0,n):

#enter student's name and score

liststudent.append(input("Enter student's name: "))

listsmark.append(int(input("Enter the marks: ")))

#display list of student and score - unsorted

print("\n")

print(liststudent)

print("unsorted list: ",listsmark)

#sort input from highest to lowest

def bubbleSort (listmark,liststudent):

for n in range(len(listmark)-1, 0, -1):

for i in range(n):

if listmark[i] < listsmark[i + 1]:

mark=listmark[i]

listmark[i]=listmark[i + 1]

listmark[i + 1]=mark

mark=liststudent[i]

liststudent[i]=liststudent[i + 1]

liststudent[i + 1]= mark

#calling bubblesort function

bubbleSort (listsmark,liststudent)

#display list of student and score - sorted

print("\n")

print(liststudent)

print("sorted list: ",listsmark)

print("\n")

#search a student and update his/her score using sequantialsearch

elif option == 2:

def sequantialSearch (liststudent,listsmark,item):

pos = 0

found = False

while pos<=len(liststudent) and not found:

if liststudent[pos] == item:

found = liststudent[pos]

print("Name searched is ", liststudent[pos],"in index", pos)

#update data

change = input("Want change his/her marks?(yes/no): ")

if change == "yes":

newValue=int(input("New value: "))

listsmark[pos]=newValue

else:

print("No update data")

else:

pos = pos + 1

return found

#enter student name

item = str(input("Enter the name of student you want to search: "))

#calling binarysearch function

sequantialSearch (liststudent,listsmark,item)

#calling bubblesort function

bubbleSort (listsmark,liststudent)

#display updated list

print("\n")

print(liststudent)

print("updated list: ",listsmark)

print("\n")

#displaying the lowest and the highest mark (with name)

elif option == 3:

#define student with the highest and lowest score

high=listsmark[0]

low=listsmark[-1]

high\_student=liststudent[0]

low\_student=liststudent[-1]

#display highest mark and lowest mark

print ("\nTHE HIGHEST MARK: ",high,"%")

print ("\nSTUDENT WITH THE HIGHEST MARK: ",high\_student)

print ("\nTHE LOW MARK: ",low,"%")

print ("\nSTUDENT WITH THE LOWEST MARK: ",low\_student)

elif option == 4:

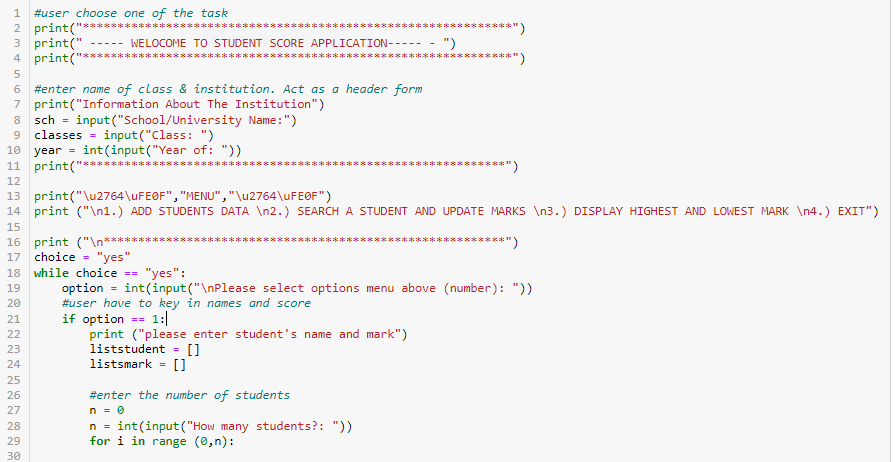
print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

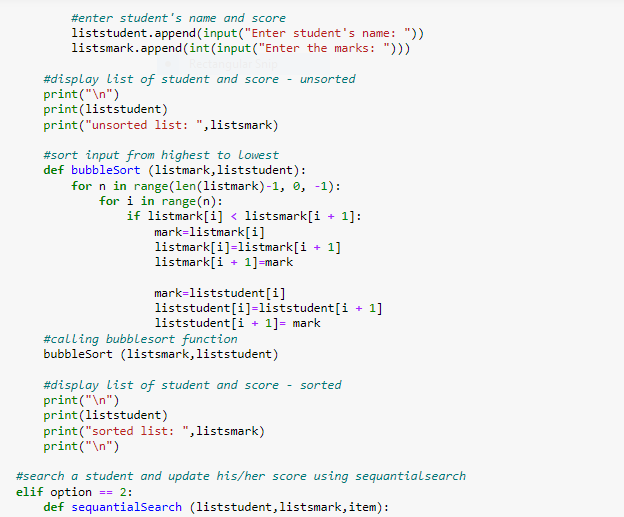
print("-------THANK YOU!!,BREAK A LEG FOR UPCOMING EXAM!------")

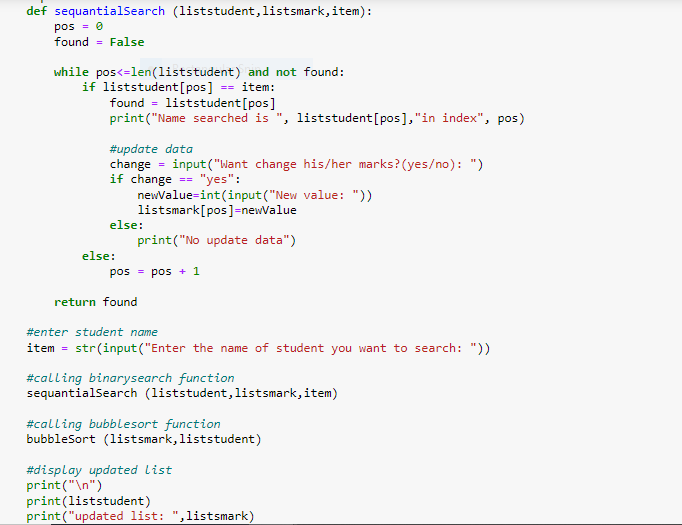
break

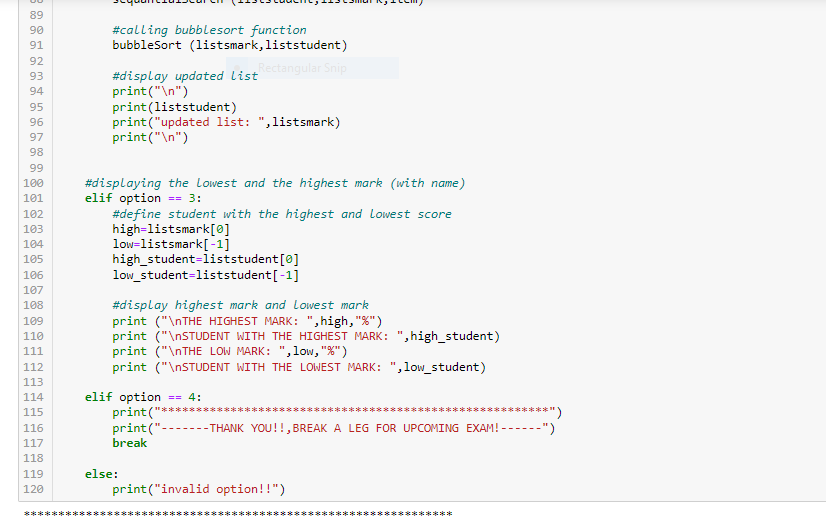
else:

print("invalid option!!")

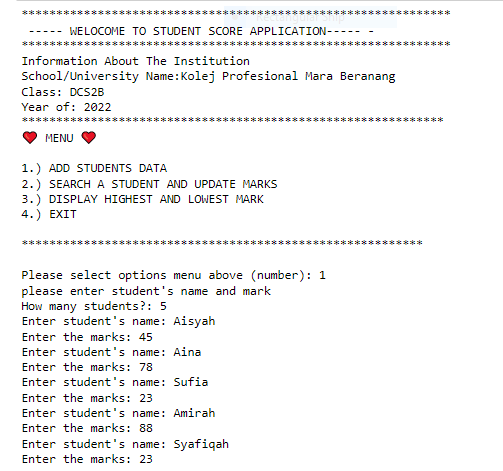


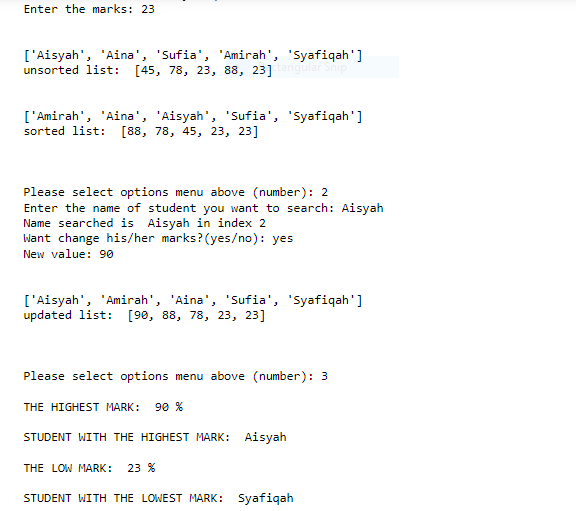


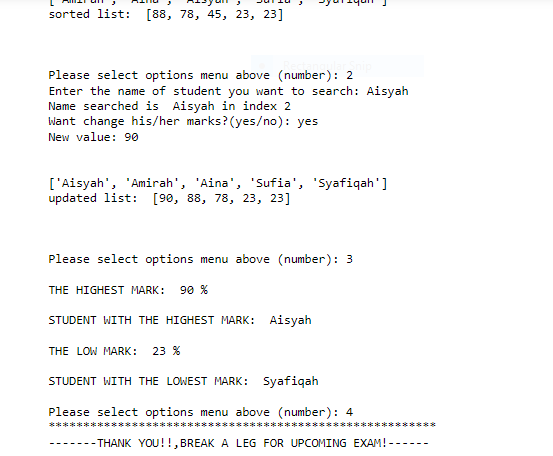




1. Output







REFERENCE LIST:

1. *Bubble sort - Sorting - KS3 Computer Science Revision*. (2020, March 2). BBC Bitesize. Retrieved February 3, 2022, from <https://www.bbc.co.uk/bitesize/guides/z2m3b9q/revision/2>
2. Kumar, A. (2017, May 19). *Sorting a list using bubble sort in Python*. CodesDope. Retrieved March 2, 2020, from <https://www.codesdope.com/blog/article/sorting-a-list-using-bubble-sort-in-python/>
3. *What is a bubble sort good for?* (2008, November 9). Stack Overflow. Retrieved January 2, 2020, from <https://stackoverflow.com/questions/276113/what-is-a-bubble-sort-good-for>
4. *Data Structure - Bubble Sort Algorithm*. (2020, March 2). Bubblesort. Retrieved February 1, 2022, from <https://www.tutorialspoint.com/data_structures_algorithms/bubble_sort_algorithm.htm>