 <b>Marwadi</b> University	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: DAA (01CT0512)</b>	<b>AIM: Selection Sort</b>	
<b>Experiment No: 02</b>	<b>Date: 25/07/2023</b>	<b>Enrolment No:</b>

### Selection Sort:

Selection Sort is another simple sorting algorithm that sorts an array by repeatedly finding the minimum element from the unsorted portion of the array and swapping it with the first unsorted element. The algorithm divides the array into two parts: the sorted part on the left and the unsorted part on the right. The sorted part grows from left to right as the smallest elements are continuously selected and placed in their correct positions.

### Algorithm:


1. Start with the first element of the array. This element is considered the initial sorted portion.
2. Find the minimum element from the unsorted portion of the array.
3. Swap the minimum element with the first unsorted element.
4. Expand the sorted portion by moving its boundary one element to the right.
5. Repeat steps 2 to 4 until the entire array is sorted.

### Code: [Paste your output here]

```
#include<iostream>
using namespace std;

void selectionsort(int arr[],int n){
    for(int i= 0; i<n-1; i++){
        for(int j = i+1; j<n; j++){
            if(arr[j]<arr[i]){
                int temp = arr[j];
                arr[j] = arr[i];
                arr[i] = temp;
            }
        }
    }
    for (int i = 0; i<n; i++){
        cout<<arr[i]<<" ";
    }
}

int main(){
    int arr[10] = {1,8,7,5,6,9,2,10,32,1};
    selectionsort(arr,10);
}
```

 <b>Marwadi</b> University	<b>Marwadi University</b> <b>Faculty of Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: DAA (01CT0512)</b>	<b>AIM: Selection Sort</b>	
<b>Experiment No: 02</b>	<b>Date: 25/07/2023</b>	<b>Enrolment No:</b>

```

        return 0;
    }

```

OUTPUT:

```

-o selectionsort } ; if ($?) { .\selectionsort }
1 1 2 5 6 7 8 9 10 32

```

Space complexity: \_\_\_\_\_

Justification: \_\_\_\_\_  
 \_\_\_\_\_

**Time complexity:**

Best case time complexity: \_\_\_\_\_

Justification: \_\_\_\_\_  
 \_\_\_\_\_

Worst case time complexity: \_\_\_\_\_

Justification: \_\_\_\_\_  
 \_\_\_\_\_