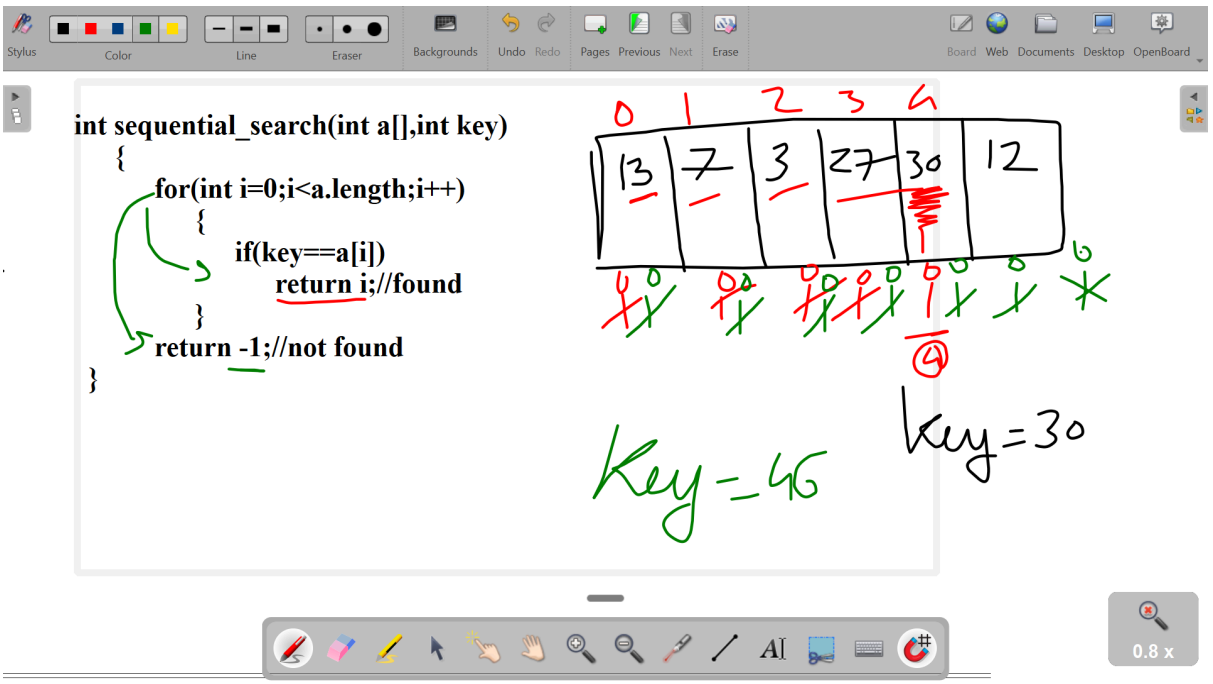



```
package Searching_Sorting;

public class Searching_Algorithms
{
   static int sequential_search(int a[],int key)
   {
      for(int i=0;i<a.length;i++)
      {
         if(key==a[i])
            return i;//found
      }
      return -1;//not found
   }
   static int binary_search(int a[],int start,int end,int key)
   {
      if(start<=end)
      {
         int mid=(start+end)/2;
         if(a[mid]==key)
            return mid;
         else
         {
            if(key<a[mid])
               return binary_search(a,start,mid-1,key);
            else
               return binary_search(a,mid+1,end,key);
         }
      }
      else
      {
```

```
            return (-1);
        }
    }

    public static void main(String[] args) {
        int a1[]={44,11,77,99,22,66};
        int a2[]={11,22,33,44,55,66,77,88,99};

    }

}

int sequential_search(int a[],int key)
        {
                for(int i=0;i<a.length;i++)
                        {
                                if(key==a[i])
                                        return i;//found
                        }
                return -1;//not found
 }
```

We start searching from the middle. (start+end)/2
Key which we are searching is "match search stops."
Key is lesser than the mid. Go to left. (start,mid-1)
He is greater than the mid. Go to right. (mid+1,end)

```
int binary_search(int a[],int start,int end)
{
  if(start<=end)
        {
                int mid=(start+end)/2;
                if(a[mid]==key)
                        return mid;
                else
                        {
                                if(key<a[mid)
                                        return binary_search(a,start,mid-1);
                                else
                                        return binary_search(a,mid+1,end);
                        }
         }
  else
        {
                return (-1);
        }
}
```

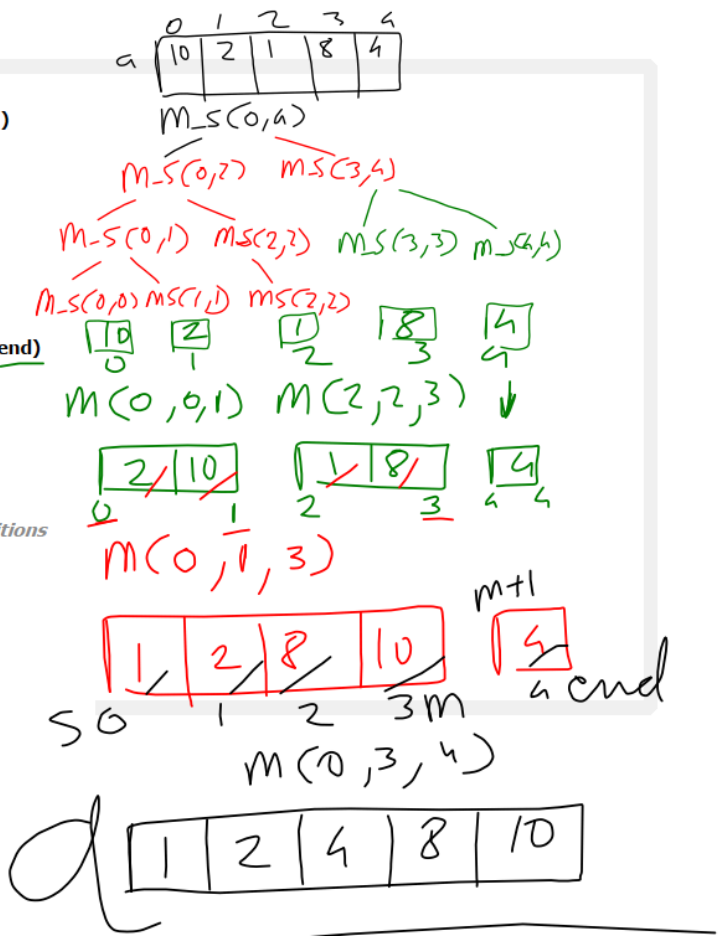
```
void optimized_bubble_sort(int a[])
        {       boolean done=ture;
                for(int i=a.length-1;i>0;i--)//limit passes
                        {
                                done=true;
                                for(int j=0;j<i;j++)//j stops at second last
                                        {
                                                if(a[j]>a[j+1])
                                                        {
                                                          int t=a[j];a[j]=a[j+1];a[j+1]=t;
                                                          done=false;
                                                        }
                                        }
                                if(done==ture)
                                        break;
                        }
```

```
package Searching_Sorting;

public class Sorting_Algorithms {
   static void optimized_bubble_sort(int a[]) {
      boolean done = true;
      for (int i = a.length - 1; i > 0; i--)//limit passes
      {
         done = true;
         for (int j = 0; j < i; j++)//j stops at second last
         {
            if (a[j] > a[j + 1]) {
               int t = a[j];
               a[j] = a[j + 1];
               a[j + 1] = t;
               done = false;
            }
         }
         if (done == true)
            break;
      }
   }
   static void print_array(int a[])
   {
      System.out.println("\nArray has:");
      for(int i:a)
      {
         System.out.print(i+", ");
      }
   }
```

```
    public static void main(String[] args) {
        int a[]={88,11,77,22,44,33,99,55,66};
        Sorting_Algorithms.print_array(a);
        Sorting_Algorithms.optimized_bubble_sort(a);
        Sorting_Algorithms.print_array(a);
    }
}



void selection_sort(int a[])
        {
                int min,pos;
                for(int i=0;i<a.length-1;i++)
                        {
                                min=a[i];pos=i;//reference
                                for(int j=i+1;j<a.length;j++)//j stops at second last
                                        {
                                                if(a[j]<min)
                                                        {//update
                                                          min=a[j];pos=j;
                                                        }
                                        }
                        //swap
                        a[pos]=a[i];a[i]=min;
                        }
}


void insertion_sort(int a[])
        {
                int element;
                for(int i=0;i<a.length-1;i++)
                        {
                                element=a[i+1];
                                j=i+1;
                                while(j>0 && a[j-1]>element)
                                        {       //move back
                                                a[j]=a[j-1];
                                                j--;//move ahead
                                        }
                        //insert
                        a[j]=element;
                        }
}
```

```
static void merge_sort(int a[],int start,int end)
   {
      if(start<end)//size-1 stop
      {
         int mid=(start+end)/2;
         merge_sort(a,start,mid);//left half
         merge_sort(a,mid+1,end);//    right half
         merger(a,start,mid,end);
      }
   }
   static void merger(int a[],int start,int mid,int end)
   {
      int i,j;//index for block
      int tindex,temp[];//temp array
      i=start;
      tindex=start;
      j=mid+1;
      temp=new int[a.length];
      while(i<=mid && j<=end)//boundry conditions
      {
         if(a[i]<a[j])
            temp[tindex++]=a[i++];
         else
            temp[tindex++]=a[j++];
      }
      while(i<=mid)
         temp[tindex++]=a[i++];
      while(j<=end)
         temp[tindex++]=a[j++];
      //copy temp to a again for round i+1
      for(i=start;i<=end;i++)
         a[i]=temp[i];
   }
```

```java
static void merge_sort(int a[],int start,int end)
{
   if(start<end)//size-1 stop
   {
      int mid=(start+end)/2;
      merge_sort(a,start,mid);//left half
      merge_sort(a,mid+1,end);//  right half
      merger(a,start,mid,end);
   }
}
static void merger(int a[],int start,int mid,int end)
{
   int i,j;//index for block
   int tindex,temp[];//temp array
   i=start;
   tindex=start;
   j=mid+1;
   temp=new int[a.length];
   while(i<=mid && j<=end)//boundry conditions
   {
      if(a[i]<a[j])
         temp[tindex++]=a[i++];
      else
         temp[tindex++]=a[j++];
   }
   while(i<=mid)
      temp[tindex++]=a[i++];
   while(j<=end)
      temp[tindex++]=a[j++];
   //copy temp to a again for round i+1
   for(i=start;i<=end;i++)
      a[i]=temp[i];
}
```



```
void quick_sort(int a[],int start,int end)
{
	int i,j,pivot,temp;
	i=start;
	pivot=start;
	j=end;
	while(i<j)
	{
		while(a[i]<a[pivot])
			i++;
		while(a[j]>a[pivot])
			j--;
		if(i<j)
			{
				temp=a[i];a[i]=a[j];a[j]=temp;
			 }
	 }
	//if pivot in start
	if(i<=end)
		quick_sort(a,i+1,end);
	//if pivot in end
	if(j>start)
		quick_sort(a,start,j-1);
}


package Searching_Sorting;

public class Sorting_Algorithms {
   static void optimized_bubble_sort(int a[]) {
      boolean done = true;
      for (int i = a.length - 1; i > 0; i--)//limit passes
      {
         done = true;
         for (int j = 0; j < i; j++)//j stops at second last
         {
            if (a[j] > a[j + 1]) {
               int t = a[j];
               a[j] = a[j + 1];
               a[j + 1] = t;
               done = false;
            }
         }
         if (done == true)
            break;
      }
   }
   static void selection_sort(int a[]) {
```

```
    int min, pos;
    for (int i = 0; i < a.length - 1; i++) {
        min = a[i];
        pos = i;//reference
        for (int j = i + 1; j < a.length; j++)//j stops at second last
        {
            if (a[j] < min) {//update
                min = a[j];
                pos = j;
            }
        }
        //swap
        a[pos] = a[i];
        a[i] = min;
    }
}



static void merge_sort(int a[],int start,int end)
{
    if(start<end)//size-1 stop
    {
        int mid=(start+end)/2;
        merge_sort(a,start,mid);//left half
        merge_sort(a,mid+1,end);//    right half
        merger(a,start,mid,end);
    }
}
static void merger(int a[],int start,int mid,int end)
{
    int i,j;//index for block
    int tindex,temp[];//temp array
    i=start;
    tindex=start;
    j=mid+1;
    temp=new int[a.length];
    while(i<=mid && j<=end)//boundry conditions
    {
        if(a[i]<a[j])
            temp[tindex++]=a[i++];
        else
            temp[tindex++]=a[j++];
    }
    while(i<=mid)
        temp[tindex++]=a[i++];
    while(j<=end)
        temp[tindex++]=a[j++];
```

```
    //copy temp to a again for round i+1
    for(i=start;i<=end;i++)
        a[i]=temp[i];
}
static void quick_sort(int a[],int start,int end)
{
    int i,j,pivot,temp;
    i=start;
    pivot=start;
    j=end;
    while(i<j)
    {
        while(a[i]<a[pivot])
            i++;
        while(a[j]>a[pivot])
            j--;
        if(i<j)
        {
            temp=a[i];a[i]=a[j];a[j]=temp;
        }
    }
    //if pivot in start
    if(i<=end)
        quick_sort(a,i+1,end);
    //if pivot in end
    if(j>start)
        quick_sort(a,start,j-1);
}




static void print_array(int a[])
{
    System.out.println("\nArray has:");
    for(int i:a)
    {
        System.out.print(i+", ");
    }
}

public static void main(String[] args) {
    int a[]={88,11,77,22,44,33,99,55,66};
    Sorting_Algorithms.print_array(a);
    //Sorting_Algorithms.optimized_bubble_sort(a);
    //Sorting_Algorithms.selection_sort(a);
    //Sorting_Algorithms.merge_sort(a,0,a.length-1);
```

```
        Sorting_Algorithms.quick_sort(a,0,a.length-1);
        Sorting_Algorithms.print_array(a);
    }
}



 static void heap_sort(int a[])
{
    int temp,i,j,pc;//pc:parent-child
    boolean done;
    for(i=a.length-1;i>0;i--)//last to first
    {
        for(j=0;j<=i;j++)
        {
            done=false;
            pc=j;
            while(pc>0 && pc/2>=0 && done!=true)
            {
                if(a[pc]>a[pc/2])
                //child   parent
                {
                    temp=a[pc];a[pc]=a[pc/2];a[pc/2]=temp;
                    pc=pc/2;//go to parent to check
                    done=false;
                }
                else
                {
                    done=true;
                }
            }
        }//j closed
        temp=a[0];a[0]=a[i];a[i]=temp;//swap largest to last
    }
}
```