Sharath NK stn9447
Rahul NG rnn4511

3a) Greedy 1

```
String Longest(int array[])
{ String result="";
  int l1=0;
  int l2 = array.length;
  while(l1<l2)
  {
   int index = getsmallestindex(array, l1, l2);
   result += Integer.toString(array[index]);
   l1 = index+1;
  }
  return result;
}
int getsmallestindex(int array[], int l1, int l2)
 {
  int current= array[l1];
  int result = l1;
  for(int i= l1; i<l2; i++)
   { if(array[i]<current)
     { current = array[i];
       result = i;
     }
   }
  return result
}
```

3b) Input : 4 5 6 7 2
Optimal Solution : 4 5 6 7
Algorithm output : 2

comments : Algorithm fails if minimum value is at last position of array

3a)
```
String longest(int array[])
{
  String result[] = new string[array.length];
  int smallest = 0;
 for(int l=0; l< array.length; l++)
    {  i = l;
      result[l] += Integer.toString(array[i]);
      for( int j=0; j<array.length; j++)
        {  if(a[i]<a[j])
            { i = j;
              result[l] += Integ.toString(array[j]);
            }
        }
 } int index = 0;
 int longest = result[0].length();
   for(int i=0; i<result.legth; i++)
     {  if (result[i].length() >longest)
          { longest = result[i].length();
            index = i;
          }
     }
    return result[index];
}
```

3b) Input : 4 5 6 1 10 7 9
Optimal : 4 5 6 7 9
Algorithm : 4 5 6 10
Comments : Algorithm fails if smaller numbers come after a big no.