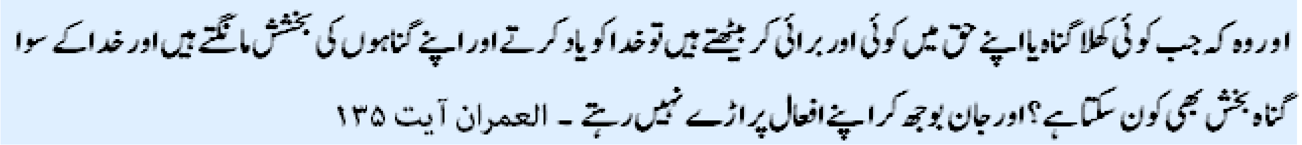

اور وہ کہ جب کوئی کھلا گناہ یا اپنے حق میں کوئی اور برائی کر بیٹھتے ہیں تو خدا کو یاد کرتے اور اپنے گناہوں کی بخشش مانگتے ہیں اور خدا کے سوا
گناہ بخش بھی کون سکتا ہے؟ اور جان بوجھ کر اپنے افعال پر اڑے نہیں رہتے ۔ العمران آیت ۱۳۵

The roots of education are bitter, but the fruit is sweet. Aristotle

Lecture 17-18

Arrays Example Function Using Arrays Sentinel Value

I am going to start with the revision of last lecture that is introduction to arrays; whereas; this lecture is a comprehensive lecture trying to cover contents of two lectures. Therefore we will discuss how to use functions with arrays and what is sentinel value? how and where to use.

Arrays are used to store list type of data. List can be of prices, marks, ages, weights, heights, scores, temperatures, topics etc. An array can be of any one type int, double, char, boolean, String etc. For a multi-column list where different columns have different items like one column has prices and second column has weights/ quantity, multiple arrays are required. There are three ways to declare an array. If we have array values available at development time we declare array like this:

```
String colors[]={"BLACK", "BLUE",...,"WHITE"};
```

List of sixteen colors given: http://www.elizabethcastro.com/html/colors/sixteencolors.html

Otherwise in both ways we have to give size of array to declare array; whereas; in above method array size is automatically equals to number of values given inside curly braces separated by comma. So a second way of declaration is:

```
String colors[]=new String[16];
```

Here we declare reference (name of array) with type. On right hand side we give keyword **new** followed by same type as given on left hand side, followed by size in square brackets. A last way is where we split declaration in two steps. In first step we declare reference only like:

```
String colors[];
```

Later we can declare array be writing reference already defined and **new** keyword followed by type and size in square brackets:

```
colors=new String[16];
```

This way is used when we want to define variables at top and declare array at later stage typically taking size as input from user. Also this way is required when we want to declare array again in the same program with different size (due to any reason) and assign to same reference because previous array values are either not required or copied into new array.

After discussing the declaration now we will learn how to use arrays after declaration with some examples. Typically we use 1 loop and an index varying from 0 to size-1 of array. For example if we have an integer type array and we want to find a value inside array:

```
int n[]={23,32,45,52,34,63,12};
int i, value=52;
for (i=0;i<n.length;i++)
     if (value==n[i])
       ?ln("Value Exist at location:"+i);
```

Similarly suppose there is an array having both negative and positive value and we want to print them in separate lines:

```
int n[]={23,-32,45,-52,-34,63,12};
int i;
?ln ("Positive Values: ");
for (i=0;i<n.length;i++)
   if (n[i]>=0)
  ? (n[i]+" ");
?ln ("Negative Values: ");
for (i=0;i<n.length;i++)
   if (n[i]<0)
      ? (n[i]+" ");
```

Another example is we have weights of players and we want to find serial no of players having weight less than average. We will do this step by step. First we will find average by calculating sum than we will check and print serial no of players having weight less than average:

```
double w[]={63.5,62.75,65.5,62,64,63,61.2,61.6,62.9,63.7};
int i;
double avg, sum=0;
for (i=0;i<w.length;i++)
   sum=sum+w[i];
avg=sum/w.length;
?ln ("Serial No of players having weight less than average:");
for (i=0;i<w.length;i++)
   if (w[i]<avg)
      ? (i+1+" ");
```

Another example is we have a list of names and we want to show name of person having name of minimum length:

```
String n[]={"Hamid Ali","Munir Ahmad","Zaheer Khan",
             "Amjad Pervaiz","Junaid Butt","Abdul Kareem"};
int i, minIndex=0;
for (i=1;i<n.length;i++)
   if (n[minIndex].length()>n[i].length())
      minIndex=i;
?ln (n[minIndex]+" has minimum length:"+n[minIndex].length());
```

Function with Arrays

Arrays can be passed to functions and received by functions question is how and why? Both questions are simple though first is conceptual and second is technical. So once again why? Answer is reusability/ modularity. Through functions we achieve two (beside other's) main advantages. One is reusability that is to do similar task again we can save rewriting code by making function and calling function whenever required. Second is modularity that we can make our code simple, clean, understandable by writing less lines doing some specific task. Therefore we can decompose our program into functions where each function has only specific code. Second question is how to use arrays with functions.

Again this is to step procedure. First is how to use them as arguments and how as parameters. Pass arrays to functions is too simple we just write name of array as argument like **`fun(a)`**; where a is name/ reference of array. Now question is how to write them as parameter this is similar to 3rd way of array declaration that is just write data type, followed by name, followed by empty square brackets (as we declare reference otherwise). See some examples:

```
     findMax(int n[])      findSum(int a[])      shuffle(String s[])
```

To pass two arrays we will write another reference in same way after comma like:

```
    compare (int a[], int b[])      subtract(double f[], int s[])
```

It is not necessary that both arrays should be of same type, they can be of any type. Let’s discuss some examples. We are writing a function to receive an array and print all values in same row:

```
public static void printRowWise(int n[]){
  for (i=0;i<n.length;i++)
    ? (n[i]+" ");
  ?ln();
}
```

Two points to be revised for students. First is function return type is independent from parameters both of them can be same or different. Secondly in above code last print statement [**?ln()**] is out of loop to move on to next line after printing all values of array. Students should remember if curly braces are not used with loop there is only 1 statement included in loop. The second statement will automatically be out of loop. Another example is function to receive integer as parameter and return sum of digits. Like 325 has sum of digits 10, 29 has 11 and so on. The idea is simple find last digit by taking remainder with 10 and drop last digit by doing integer division again with 10:

```
public static int sumOfDigits(int n){
  int i,sum=0;
  while(n>0){
    sum=sum+n%10;
    n=n/10;
  }
  return sum;
}
```

Another example is to find maxAlphabet from name as String:

```
public static char maxAlphabet(String s){
  int i;
  char maxAlphabet=s.charAt(0);
  for(i=1;i<s.length();i++)
    if (maxAlphbet<s.charAt(i))
      maxAlphabet=s.charAt(i);
  return maxAlphabet;
}
```

Another example is to add some value in all elements of array:

```
public static void addValue(int n[], int k){
  for(i=0;i<n.length;i++)
    n[i]=n[i]+k;
}
```

Note we are not printing nor returning in above function. Students should understand that here we are adding value in array elements and this change is in array not in function therefore array values will automatically change wherever array is created and passed to function but here we are concluding this topic by calling this method and another method that is to print array values row wise:

```
 public static void main(String []ar){
  int m[]={23,34,26,36,41,16};
  printRowWise(m);
  addValue(m, 10);
  printRowWise(m);
}
```

Just write all above function in single class and execute the code to see how it is working.

Sentinel Value

Array is typically a single column list, we may have some values empty in the list. For example we have prices of items and prices of some items is not available so far that does not mean to remove them, because they may become available later. Similarly suppose there is list of marks of quiz where some students were absent in the quiz and we can't remove them to keep sequence of list. So how to handle missing values. There is concept of sentinel value that is any value other than possible data values. For example for marks of students -1 can be a sentinel value, same is for score of players. However -1 is not always available, like if we have temperature of cities where some cities may have temperature below 0. Therefore we may use 100 either in +ive or –ive as sentinel value because this is beyond any possible temperature of cities. Let's consider example of score of players in a match. Again 0 score doesn't mean that a player has not bat in the match, it is possible that player is out without score:

```
class ArrayExample1{
   public static void main(String []args){
     int scores[]={23,0,16,-1,45,0,18,-1,-1,-1};
     System.out.println("     Score Card");
     for (i=0;i<scores.length;i++){
        System.out.print("Player 1:\t");
        if (scores[i]==-1)
           System.out.print("Do Not Bat");
        else
           System.out.println(scores[i]);
   }
}
```

In above example printing -1 is meaning less for user; whereas ; message Do Not Bat is understandable by any person. Similarly we may use multiple values as sentinel values like -2 for retired hurt etc.