Development of the randomization mechanism for online and offline collaborative marking

Ozuomba Simeon[1]
Department Of Electrical/Electronic And Computer Engineering, University of Uyo, Akwa Ibom State Nigeria

Queenta Edem Akpan[2]
Department Of Electrical/Electronic And Computer Engineering, University of Uyo, Akwa Ibom State Nigeria

Phillip M. Asuquo[3]
Department Of Electrical/Electronic And Computer Engineering, University of Uyo, Akwa Ibom State Nigeria

***Abstract*— In this paper, development of randomization mechanism for online and offline collaborative marking system is presented. Collaborative marking mechanism enables a course lecturer with large class population to use the students to mark the quiz and continuous assessment scripts. The program is coded in C programing language. The program design is based on random number generator and swap functions which are applied to the randomization array used to hold the numbers for identifying each of the students and for shuffling the numbers to ensure that no student is assigned his script to mark. More so, in the offline scenario, the student's script is not allowed also to be assigned to the immediate neighbors sitting on the left or right of the student in the same row. The same shuffling output when applied to the in-between row allocation ensures that no student script is assigned to the same row of the student and in the offline case, no script is assigned to the rows immediately in front or back of the student. The program validates the shuffling output by using a validation output array in which a '0' in the output means a student is assigned his or her script and a '1' or '-1' means a student's script is assigned to the immediate neighbors of the students. The online case is meant to avoid the presences of '0' in the validation array output and the offline case is mean to avoid the presence of '0', '1' and '-1' in the validation output. The sample numerical examples done with a class of 200 students showed that the program satisfied the conditions in the online and the offline scenarios.**

Keywords*— *Randomization, Online Learning Platform, Random Number Generator, Offline or Face-To-Face Learning, Collaborative Marking System, Web Application, C Program

1. Introduction

Quality education has been identified as one of the key strategies that will enable the developing countries to strive and catch up with the developed nations [1,2,3,4,5, 6,7,8,9,10, 11,12,13,14, 15,16, 17,18,19,20,21]. Intentional and focus-driven education with emphases on science and technologies has been advocated in many quarters [22,23,24,25,26,27,28,29,30,31,32,33,34,35]. However, the lack of requisite infrastructure and qualified manpower in the educational institutions pose additional problem [36,37,38,39,40,41,42,43,44,45,46,47].

One of the recurring problems in the tertiary institutions is the high students-to-staff ratio which means that one staff is meant to attend to large number of students at a time [48,49,50,51,52,53,54, 55,56,57,58,59]. The implication is that effective attention and commitment to the individual student is lacking. Moreover, effective continuous assessment of the students is difficult to implement as each quiz or examination will take substantial amount of the lecturer's time to mark for such large classes. In such cases, the requisite feedback such continuous assessment affords the lecturer for effective delivery of the course materials is unfortunately missing. As such, in this paper, collaborative marking mechanism that enables a course lecturer with large class population to use the students to mark the quiz and continuous assessment scripts a presented. The mechanism is designed for two different scenarios, namely, online case where the students submit their script through an online platform and they are assigned the scripts through the online platform. In this case, physical presence and contact is not expected. In the offline collaborative case, the students are physically present in the class and seated in rows and columns. That means, physical answer booklets are assigned to the students who are physically present in the class.

The mechanism operates by shuffling the answer scripts and assigning them to the students such that no student is assigned his or her script to mark. This condition satisfies the online collaborative mechanism. In the offline case, additional requirements is that the immediate neighbors of student are not assigned the student's script to mark. In this case, the immediate neighbors of student are students sitting on the left and right hand sides of the student. The design for the two cases are related but with few differences. The offline case is a bit more complex than the online case.

In all, the mechanism rely on two major programing functionalities, random number generator and number swapping functions. The two functions are applied on array cells that are used to hold the integer numbers that identifies each of the students in the class. The program is implemented in C language. It also provides a way of validating the effectiveness of the randomization output to know if the requisite conditions are satisfied for the offline or online scenario.

2. Methodology

The design is presented the two different scenarios, the online and the offline mechanisms. The online case is presented first and the modification of the design to suit the offline case is then presented.

2.1 Design of the randomization mechanism for online collaborative marking system

The entire mechanism relies on two major programing functionalities, namely, random number generation and swapping of memory locations specified in array. In this paper, the entire program is decomposed into modules. The program is implemented in C programing languages where function is the major building block. As such, in the design approach adopted in the study, the function is used to denote a program segment that function as a unit and delivers a specific function, such as random number generation, swapping of two array cell contents, printing the content of array. On the other hand, a module in this study is used to describe a program segment that consist of sequence of statements and function calls that is used to achieve a group of related functionalities of the program. For instance, in the mechanism presented in this paper, a module for initialization of the randomization array requires iterative process of repeated initialization of the array content such that array content undergoes repeated process of assigning integer values to the array cells, calling on random number generation function to generate random number for each array cell, calling on the swap function for swapping the array cell content, calling on the array printing function to display the current array content, repeating the process until all the array cell contents are randomized by changing their contents based on the generated random number. Next, the module repeats the array randomization by using the last randomized array content as the initial array values for the randomization process.

Specifically, the major modules used in the randomization mechanism for online collaborative marking system includes:

Module 1: Initialize variables and select number of students for the randomization process

Module 2: Initialization of randomization array

Module 3: Repeated randomization and display of randomization output until no more randomization is needed

Module 4: Validate the contents of the randomization output array by calling on the validation function

The functions used in used in the randomization mechanism for online collaborative marking system includes:

Function 1: Function for printing the output from the randomization array

Function 2: Function for swapping the content of two cell locations in the randomization array

Function 3: Function for generating the random number for each of the cells in the randomization array

Function 4: Function for printing the output from the randomization output array

Function 5: Function for generating and printing the output from the randomization validation output array

2.1.1 Module 1: Initialize variables and select number of students for the randomization process

The various global variable and arrays used in the program are initialized. The number of students (denoted as Nmx) that are required for the randomization process is read into the system. The variable, Nmx can be used to represent the total number of students (denoted as TNmx) that submitted their scripts for collaborative marking. In this case, since TNmx = Nmx, the randomization array (denoted as ArrNm[Nmx]) is a one dimensional array that contains all the TNmx students in the class that submitted their scripts for marking. Also, the output array, (denoted as AopNm[Nmx][1]) is a two dimensional array that has a single row and contains all the students in the class that submitted their scripts for marking.

Alternatively, the variable, Nmx can be used to represent the total number of students that are grouped as a row of students, as if the students are sitting in rows in a physical classroom. In this case, since TNmx > Nmx, and the randomization array (denoted as ArrNm[Nmx]) is a one dimensional array that contains all the TNmx students in the class that submitted their scripts for marking row. Also, the output array, (denoted as AopNm[Nmx][Kount] where Kount =1,2,3,…KountMx) is a two dimensional array that has KountMx rows and contains all the students in the class that submitted their scripts for marking, but segmented into KountMx rows, where;

$$\text{KountMx}\begin{cases}\geq \left\lfloor\left(\frac{\text{TNmx}}{Nmx}\right)+1\right\rfloor & \text{for TNmx} > \text{ Nmx}\\ = 1 \quad \text{for TNmx} = \text{ Nmx}\end{cases} \tag{1}$$

So, once TNmx and Nmx are known, then KountMx is determined. In the actual implementation, Nmx is read in and TNmx is determined from the number of randomization iterations, KountMx which the user can determine from the randomization round counter (Knount). In that case, The Kount starts from 1 and increases each time the randomization process is iterated. So, the user stops when it is certain that equation for KountMx is satisfied. In this case, the user can determine the value of TNmx as follows;

$$\text{TNmx} = \text{ KountMx}\,(Nmx) \tag{2}$$

When TNmx is not exact multiples of Nmx, there will be a remainder, Rmx given as follows;

$$\text{Rmx} = \text{TNmx} - \left\lfloor\left(\frac{\text{TNmx}}{Nmx}\right)\right\rfloor \tag{3}$$

The user can randomize the Rmx students by running the program a second time with Nmx = Rmx and KountMx =1.

2.1.2 Module 2: Initialization of randomization array

At this point, the randomization array (denoted as ArrNm[Nmx]) is first initialized by assigning ArrNm[x] =x for x =1,2,3,…Nmx, that is;

$$\text{ArrNm}[x] = x \ \ where x = 1,2,3, \dots Nmx \qquad (4)$$

Also, the randomization initialization counter (denoted as KountX) is initialized to 0, that is $\text{KountX} = 0$.Then, the random number generator function (denoted as FnRandGen) is used to generate random number for each of the cell locations in the ArrNm[x] array. For each array index, x the FnRandGen function is called with lower value of 1 and upper value of Nmx and the condition that the returned value of the random number (denoted as RanX) must not be the same as x.

When RanX is obtained, the swap function (denoted as FnSwap) is called to swap the content of the array cell ArrNm[x] with the content of array cell ArrNm[RanX]. The process is repeated for x =1, 2, 3,…Nmx. This forms one complete randomization of the randomization array ArrNm[x]. The function (denoted as FnOPArrNm) is called. The function FnOPArrNm is for printing the output from the randomization array, ArrNm[x] x =1, 2, 3,…Nmx. At this point, the randomization initialization counter (denoted as KountX) is increased by one, that is;

$$\text{KountX} = \text{KountX} + 1 \qquad (5)$$

The present randomization array content is used as the initial value for the next ($\text{KountX} + 1$) randomization process. The iteration is performed for a set maximum number. In this paper, the maximum value is 10. Hence, the randomization initialization process stops when KountX is at least equal to 10, that is, whenKountX ≥ 10.

2.1.3 Module 3: Repeated randomization and display of randomization output until no more randomization is needed

The last contents of ArrNm[x] are stored as the first row in the randomization output array, that means,

$$\left.\begin{array}{l} \text{KountX} = 10 \\ Kount = 1 \\ AopNm[Kount][x] = ArrNm[x] \end{array}\right\} \qquad (6)$$

The function (denoted as FnOPAopNm) is called. The function FnOPAopNm is for printing the output from the randomization output array, $AopNm[Kount][x]$ for the current $Kount$ and for x =1, 2, 3,…Nmx.

At this point, the randomization round counter (Knount) is increased by one, that is;

$$\text{Kount} = \text{Kount} + 1 \qquad (7)$$

The present randomization array content is used as the initial value for the next ($\text{Kount} + 1$) randomization process. The contents of ArrNm[x] array are stored as the $\text{Kount} + 1$ row in the randomization output array, $AopNm[Kount][x]$. The function (denoted as FnOPAopNm) is called to print the output from the randomization output array, $AopNm[Kount][x]$ for the current $Kount$ and for x =1, 2, 3,…Nmx. The iteration is repeated until $Kount$= KountMx.

2.1.4 Module 4: Validate the contents of the randomization output array by calling on the validation function

The condition for effective randomization is that a student, denoted by the index number, x should not be assigned his script to mark, where the script number if the content of the randomization array, ArrNm[x]. Hence, the condition is that at the end of the randomization process, then;

$$\text{ArrNm}[x] \neq x \ \ for \ x = 1,2,3, \dots Nmx \qquad (8)$$

This can easily be proven analytically as follows;

$$x - \text{ArrNm}[x] \neq 0 \ \ for \ x = 1,2,3, \dots Nmx \qquad (9)$$

So, the validation results is examined to check if there is any zero (0) in the validation output. Specifically, the validation process is implemented with respect to the randomization output array, $AopNm[Kount][x]$. At this point, the function (denoted as FnValAopNm) is called to generate and print validation output from the randomization validation output array, where the validation output is generated from $AopNm[Kount][x]$ as follows;

$$x - AopNm[Kount][x], \ \ for\ each\ Kount\ and\ \ x = 1,2,3, \dots Nmx \qquad (10)$$

Where $Kount$ = 1,2,3,…, $KountMx$. As long as there is no zero (0) in the validation output, the randomization process in certified acceptable.

2.1.5 The Random number generator function

In C programing, the random number generator function is rand () and it can be used to generate random integer number in the range from Lower to Upper, where Lower is the lowest number specified by the user and Upper is the highest number specified by the user . So, the output of rand() function will be within the range with Lower and Upper values included as possible output values. The C program syntax for rand(0 function is as follows;

```
RandNum = (rand( ) % (upper - lower + 1)) + lower
```

Where RandNum is the generated random number such that Lower ≤ RandNum ≤ Upper.

Essentially, the basic parameters needed in the function call for the rand() are Lower and Upper. In the case of the collaborative mechanism program, the Lower is usually 1 and the Upper is Nmx. Also, additional parameter needed for the rand() is the current value of the array index, denoted as x which is used to specify the condition that the rand() be repeated until RandNum $\neq x$. Hence, in the program design for the online collaborative script marking platform, the rand() function has four parameters and the function is specified in C language as follows;

```
int rand( int Lower, int Upper, int x)
{ int RandNum =x;
While   (RandNum ==x) { RandNum = (rand( ) % (upper - lower + 1)) + lower ;};
return (RandNum)
}
```

The function call to the rand( ) in this paper is usually as follows;

```
RanX = rand(   1, Nmx, x);
```

The program segment entails that the RandNum is reputedly computed as long as RandNum =x. The calculation stops when RandNum ≠x and the value of RandNum is returned as the generated random number for the randomization array cell with index number, x. The value of $1 \leq$ RandNum $\leq$ Nmx.

At this pint, the swap function is called to swap the content of the randomization array cell x with the content of the randomization array cell RandNum.

2.1.6 The swap function

The swap function is used in this program to swap the contents of two array cell with index number x and y. If, for instance, the array is the randomization array, then the two array cell contents are ArrNm[x] and ArrNm[y] and the swap function is implemented as follows;

```
int  Swap ArrNm (int x, int y)
{ int num;
num  = ArrNm[x]  ;
ArrNm[x]    = ArrNm[y]  ;
ArrNm[y]    = num  ;
Return;
}
```

In this case, it is assumed that the array = ArrNm[x] is a global variable that is accessible to the function Swap( ). So, after the random number RandNum is generated for the array cell index x, the swap function is called as follows;

```
Swap ArrNm (x, RandNum) ;
```

The function will swap the contents of the randomization array cell locations x and RandNum.

2.1.7 Function for printing the output from the randomization array

The For loop is used to print or display the contents of an array for index 1 to array index Nmx where Nmx is the highest array index specified in the array declaration or desired by the user at the time of printing. For instance, in order to display the contents of the randomization array from cell index 1 to cell index Nmx, the following C program segment can be used;

```
for(x=1 ; x<= Nmx ; x++){ printff("%d, ArrNm[x]);};
```

Other formatting specifications can be applied to design the layout of the output from the array print function.

2.2 The design of the randomization mechanism for offline collaborative marking system

In the case of offline collaborative marking mechanism, the students are physically present in the class and it is assumed they are seated in rows of Nmx students per row. The randomization process is done such that:

(i) A student x does not get his script assigned to him/her. That means, as earlier stated;

$$\text{ArrNm}[x] \neq x \;\; for\; x = 1,2,3, \dots Nmx \tag{11}$$

Which can be generally expressed in absolute number format as;

$$|\text{x} - \text{ArrNm}[\text{x}]| > 0 \;\; for\; x = 1,2,3, \dots Nmx \tag{12}$$

In this case, the validation results will not have any zero (0) value. However, 1 to Nmx and –Nmx to -1 are permitted in the validation output.

(ii) A student x does not get his script assigned to him/her immediate neighbors. That means, the script of student with index number x cannot have his/her script assigned to x-1 of x+1. Hence,

$$|\text{x} - \text{ArrNm}[\text{x}]| > 01 \;\; for\; x = 1,2,3, \dots Nmx \tag{13}$$

In this case, the validation results will not have any one (1) or minus one (-1) value. However, 2 to Nmx and –Nmx to -2 are permitted in the validation output.

This is achieved by some modifications on the conditions for the random number generator function. Notably, for each array index, x the FnRandGen function is called with lower value of 1 and upper value of Nmx and the condition that the returned value of the random number (denoted as RanX) must not be the same as x (that is RanX≠ x), RanX≠ x+1 and RanX≠ x-1. RanX fails any of the three conditions, then the RanX is discarded and the the FnRandGen function is called repeatedly until the three conditions are met simultaneously. Another important condition is that if the cell number x has already been swapped by the time it got to the turn of cell x to get the FnRandGen function output, then, the cell x is skipped and the next cell x +1 is examined and attended to for randomization and swapping. Specifcally, the rand( ) for the offline scenario is implemented as follows;

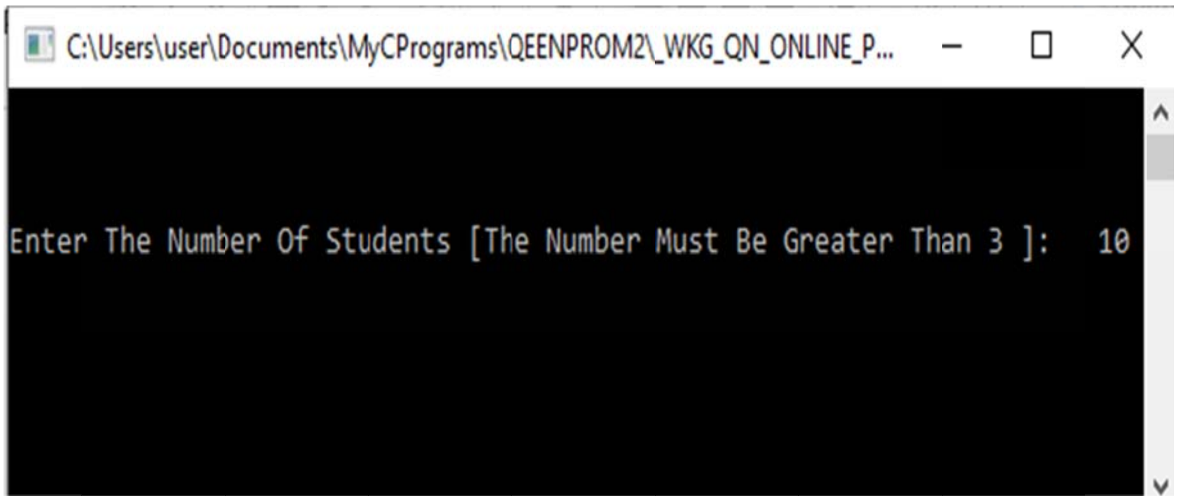
```
int  rand( int Lower, int Upper, int x)
{ int RandNum =x;
    If (x <=1) {xLw =1;} else {x=x-1;};
    If (x >=Upper) {xUp=Upper;} else
                {xUp=x+1;};
 While  (RandNum ==x && RandNum
       !=xLw && RandNum !=xUp) {
 RandNum = (rand( ) % (upper - lower
                + 1)) + lower ;};
           return (RandNum)
}
```

3. Results and Discussion

The C program was implemented for the online and offline collaborative marking systems. In each of them, two cases where considered, (i) a class of 200 students arranged in 20 rows (ii) a class of 200 students all randomized as one row
.

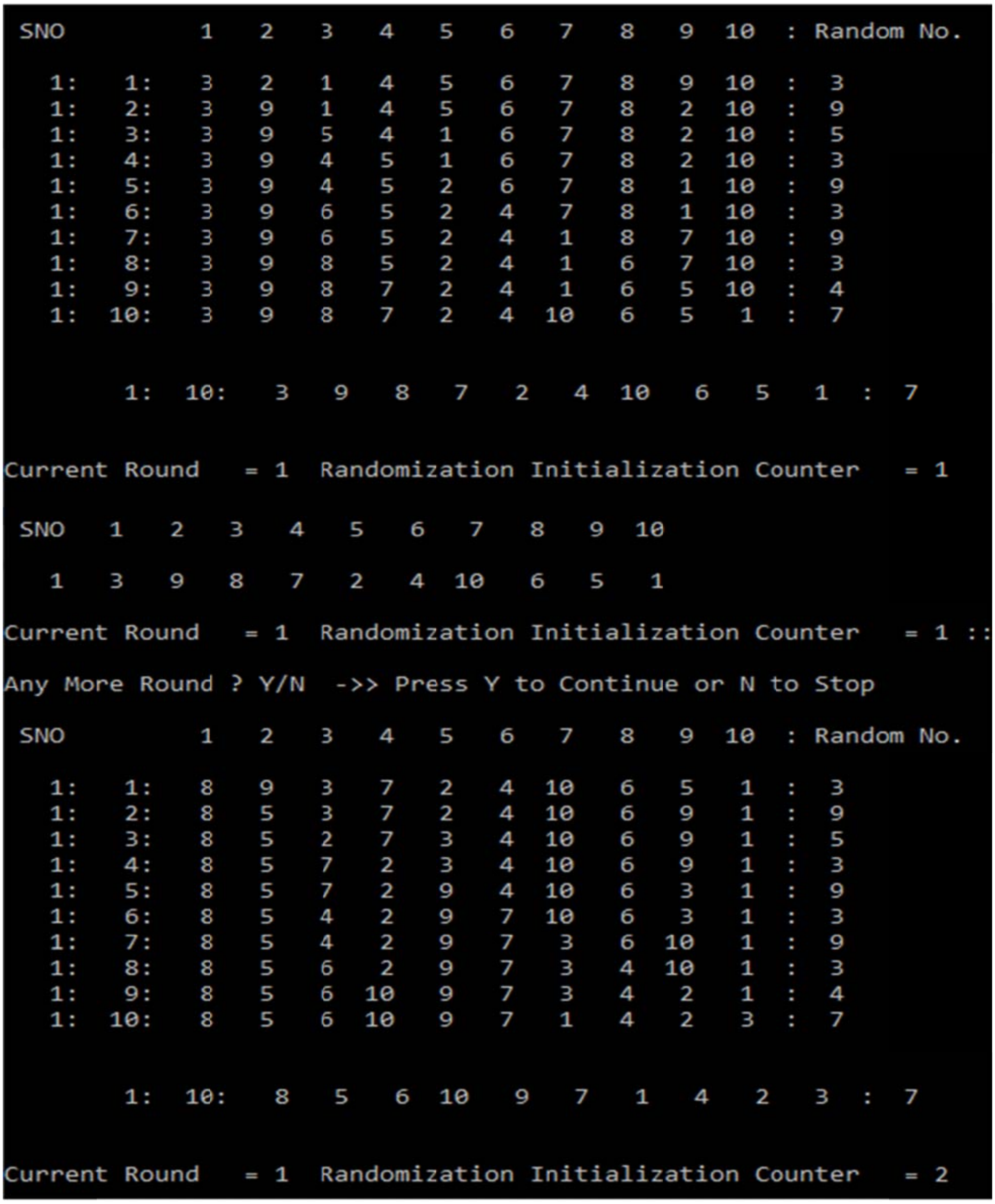
3.1 Results of the randomization mechanism C program for the online collaborative marking system: Case 1 with a class of 200 students arranged in 20 rows of 10 students per row

The results of the randomization mechanism C program for the online collaborative marking system is presented in Figure 1 to Figure 5 for the case 1 with a class of 200 students arranged in 20 rows of 10 students per row and in Figure 6 to Figure 8 for the case 2 with a class of 200 students all randomized as one row. In Figure 1 is shown the screenshot of the program output for setting the number of students considered in the collaborative marking. In this case, 200 students are expected but they are to be handled in 20 row of 10 students per row, so, KountMx =10. Hence, the needed value on the input line is 10, as shown in Figure 1. The screenshot of the program output for the first and second rounds of the initialization of the randomization array are shown in Figure 2. The screenshot in Figure 2 shows that in each initialization counter number, the initialization array goes through KountMx (in this case, 10) iterations and in each iteration, the random number generated is indicated on the rightmost column. The screenshot of the program output for the 20 rows generated and stored in the randomization output array is given in Figure 3, the validation request screenshot is shown in Figure 4 while the screenshot of the program output for the validation results of the 20 rows generated and stored in the randomization output array is shown in Figure 5. It can be seen from Figure 5 that there is no ‘0’ in the validation output array for the 20 rows of 10 per row. Again, the total number in the output of Figure 3 and Figure 5 is 200 which corresponds to the total number of students considered in this case. The results show that in the first row (row with SNO =1), student 1 is assigned the script of student 2, student 2 is assigned the script of student 10 and student 3 is assigned the script of student 9. Similarly, in the 20th row (row with SNO =20), student 1 is assigned the script of student 9, student 2 is assigned the script of student 10 and student 3 is assigned the script of student 7. Accordingly, in Figure 5, there is no ‘0’ in the output which shows that the randomization process is acceptable; no student will be assigned his script to mark.


C:\Users\user\Documents\MyCPrograms\QEENPROM2_WKG_QN_ONLINE_P...

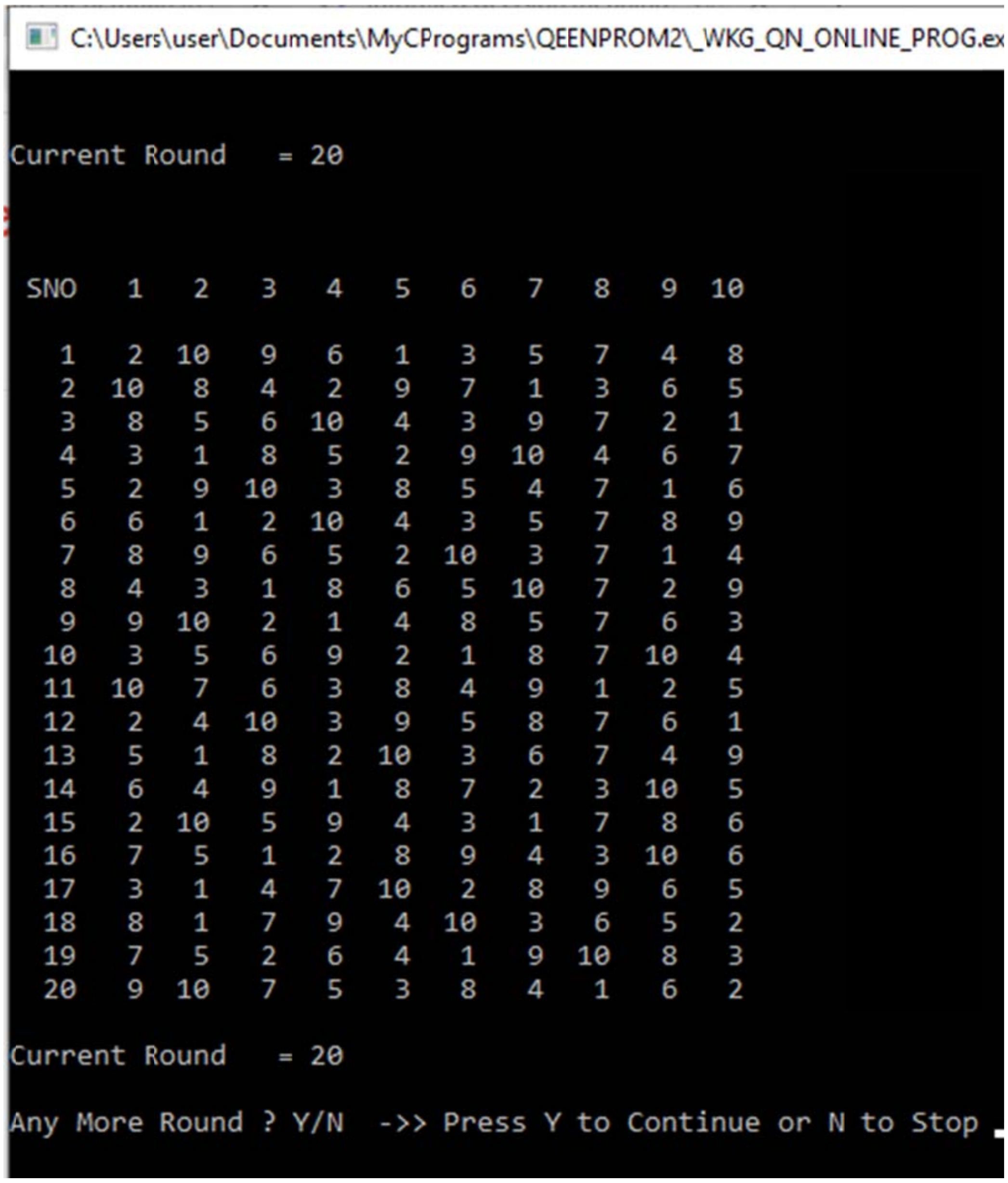
```
Enter The Number Of Students [The Number Must Be Greater Than 3 ]:   10
```

Figure 1 The screenshot of the program output for setting the number of students considered in the collaborative marking



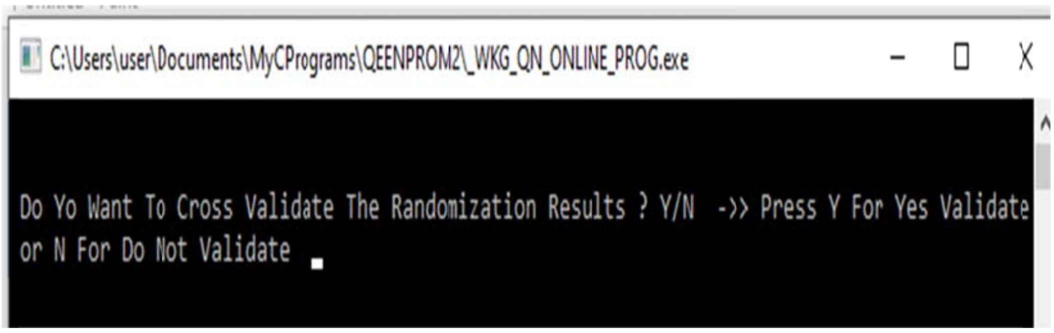
```
 SNO          1   2   3   4   5   6   7   8   9  10  : Random No.

   1:   1:    3   2   1   4   5   6   7   8   9  10  :  3
   1:   2:    3   9   1   4   5   6   7   8   2  10  :  9
   1:   3:    3   9   5   4   1   6   7   8   2  10  :  5
   1:   4:    3   9   4   5   1   6   7   8   2  10  :  3
   1:   5:    3   9   4   5   2   6   7   8   1  10  :  9
   1:   6:    3   9   6   5   2   4   7   8   1  10  :  3
   1:   7:    3   9   6   5   2   4   1   8   7  10  :  9
   1:   8:    3   9   8   5   2   4   1   6   7  10  :  3
   1:   9:    3   9   8   7   2   4   1   6   5  10  :  4
   1:  10:    3   9   8   7   2   4  10   6   5   1  :  7


        1:  10:   3   9   8   7   2   4  10   6   5   1  :  7


Current Round   = 1  Randomization Initialization Counter   = 1

 SNO   1   2   3   4   5   6   7   8   9  10

   1   3   9   8   7   2   4  10   6   5   1

Current Round   = 1  Randomization Initialization Counter   = 1 ::

Any More Round ? Y/N  ->> Press Y to Continue or N to Stop

 SNO          1   2   3   4   5   6   7   8   9  10  : Random No.

   1:   1:    8   9   3   7   2   4  10   6   5   1  :  3
   1:   2:    8   5   3   7   2   4  10   6   9   1  :  9
   1:   3:    8   5   2   7   3   4  10   6   9   1  :  5
   1:   4:    8   5   7   2   3   4  10   6   9   1  :  3
   1:   5:    8   5   7   2   9   4  10   6   3   1  :  9
   1:   6:    8   5   4   2   9   7  10   6   3   1  :  3
   1:   7:    8   5   4   2   9   7   3   6  10   1  :  9
   1:   8:    8   5   6   2   9   7   3   4  10   1  :  3
   1:   9:    8   5   6  10   9   7   3   4   2   1  :  4
   1:  10:    8   5   6  10   9   7   1   4   2   3  :  7


        1:  10:   8   5   6  10   9   7   1   4   2   3  :  7


Current Round   = 1  Randomization Initialization Counter   = 2
```

Figure 2 The screenshot of the program output for the first and second rounds of the initialization of the randomization array



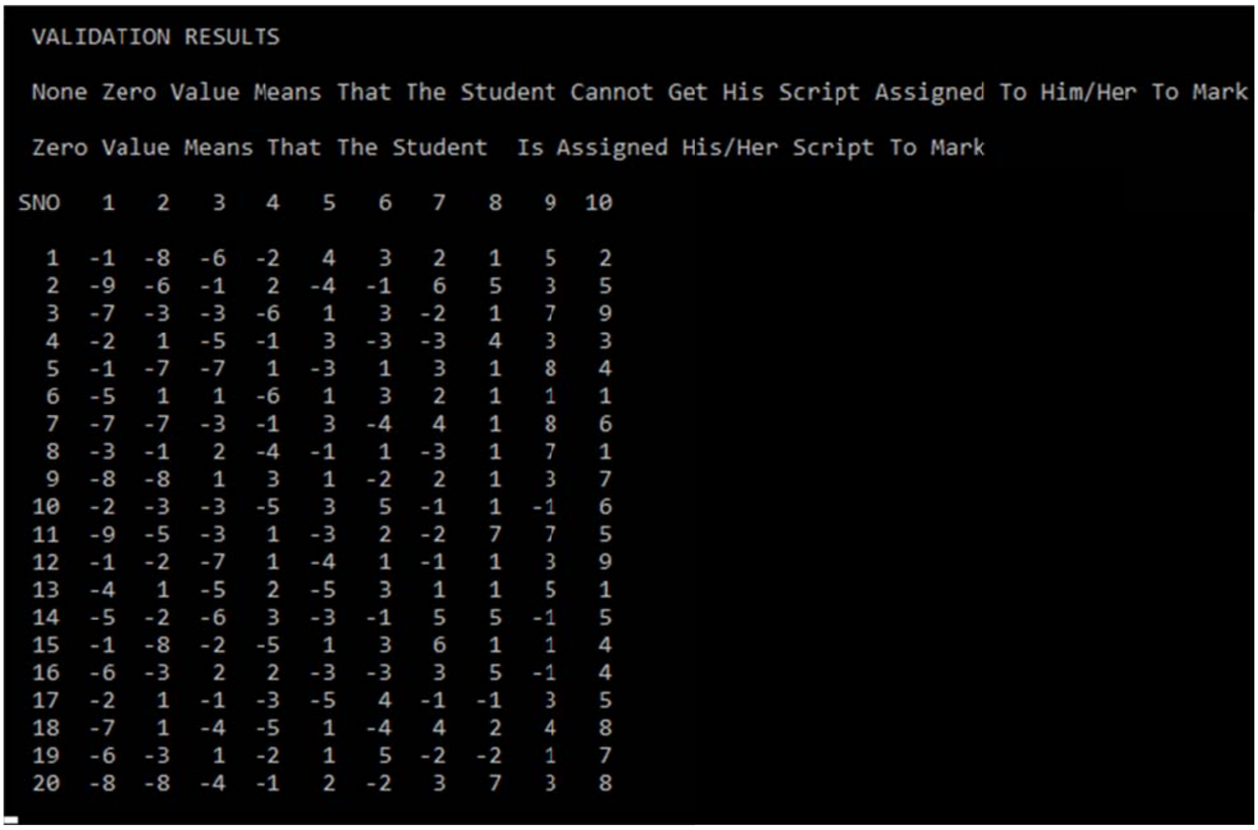
```
C:\Users\user\Documents\MyCPrograms\QEENPROM2\_WKG_QN_ONLINE_PROG.ex

Current Round    = 20


 SNO   1   2   3   4   5   6   7   8   9  10

   1   2  10   9   6   1   3   5   7   4   8
   2  10   8   4   2   9   7   1   3   6   5
   3   8   5   6  10   4   3   9   7   2   1
   4   3   1   8   5   2   9  10   4   6   7
   5   2   9  10   3   8   5   4   7   1   6
   6   6   1   2  10   4   3   5   7   8   9
   7   8   9   6   5   2  10   3   7   1   4
   8   4   3   1   8   6   5  10   7   2   9
   9   9  10   2   1   4   8   5   7   6   3
  10   3   5   6   9   2   1   8   7  10   4
  11  10   7   6   3   8   4   9   1   2   5
  12   2   4  10   3   9   5   8   7   6   1
  13   5   1   8   2  10   3   6   7   4   9
  14   6   4   9   1   8   7   2   3  10   5
  15   2  10   5   9   4   3   1   7   8   6
  16   7   5   1   2   8   9   4   3  10   6
  17   3   1   4   7  10   2   8   9   6   5
  18   8   1   7   9   4  10   3   6   5   2
  19   7   5   2   6   4   1   9  10   8   3
  20   9  10   7   5   3   8   4   1   6   2

Current Round    = 20

Any More Round ? Y/N  ->> Press Y to Continue or N to Stop _
```

Figure 3 The screenshot of the program output for the 20 rows generated and stored in the randomization output array



```
C:\Users\user\Documents\MyCPrograms\QEENPROM2\_WKG_QN_ONLINE_PROG.exe


Do Yo Want To Cross Validate The Randomization Results ? Y/N  ->> Press Y For Yes Validate
or N For Do Not Validate _
```

Figure 4 The screenshot requesting for validation process

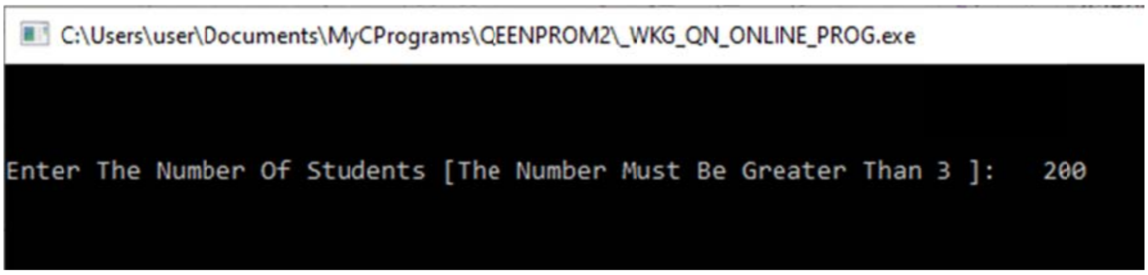


```
 VALIDATION RESULTS

 None Zero Value Means That The Student Cannot Get His Script Assigned To Him/Her To Mark

 Zero Value Means That The Student  Is Assigned His/Her Script To Mark

SNO   1   2   3   4   5   6   7   8   9  10

  1  -1  -8  -6  -2   4   3   2   1   5   2
  2  -9  -6  -1   2  -4  -1   6   5   3   5
  3  -7  -3  -3  -6   1   3  -2   1   7   9
  4  -2   1  -5  -1   3  -3  -3   4   3   3
  5  -1  -7  -7   1  -3   1   3   1   8   4
  6  -5   1   1  -6   1   3   2   1   1   1
  7  -7  -7  -3  -1   3  -4   4   1   8   6
  8  -3  -1   2  -4  -1   1  -3   1   7   1
  9  -8  -8   1   3   1  -2   2   1   3   7
 10  -2  -3  -3  -5   3   5  -1   1  -1   6
 11  -9  -5  -3   1  -3   2  -2   7   7   5
 12  -1  -2  -7   1  -4   1  -1   1   3   9
 13  -4   1  -5   2  -5   3   1   1   5   1
 14  -5  -2  -6   3  -3  -1   5   5  -1   5
 15  -1  -8  -2  -5   1   3   6   1   1   4
 16  -6  -3   2   2  -3  -3   3   5  -1   4
 17  -2   1  -1  -3  -5   4  -1  -1   3   5
 18  -7   1  -4  -5   1  -4   4   2   4   8
 19  -6  -3   1  -2   1   5  -2  -2   1   7
 20  -8  -8  -4  -1   2  -2   3   7   3   8
```

Figure 5 The screenshot of the program output for the validation results of the 20 rows generated and stored in the randomization output array

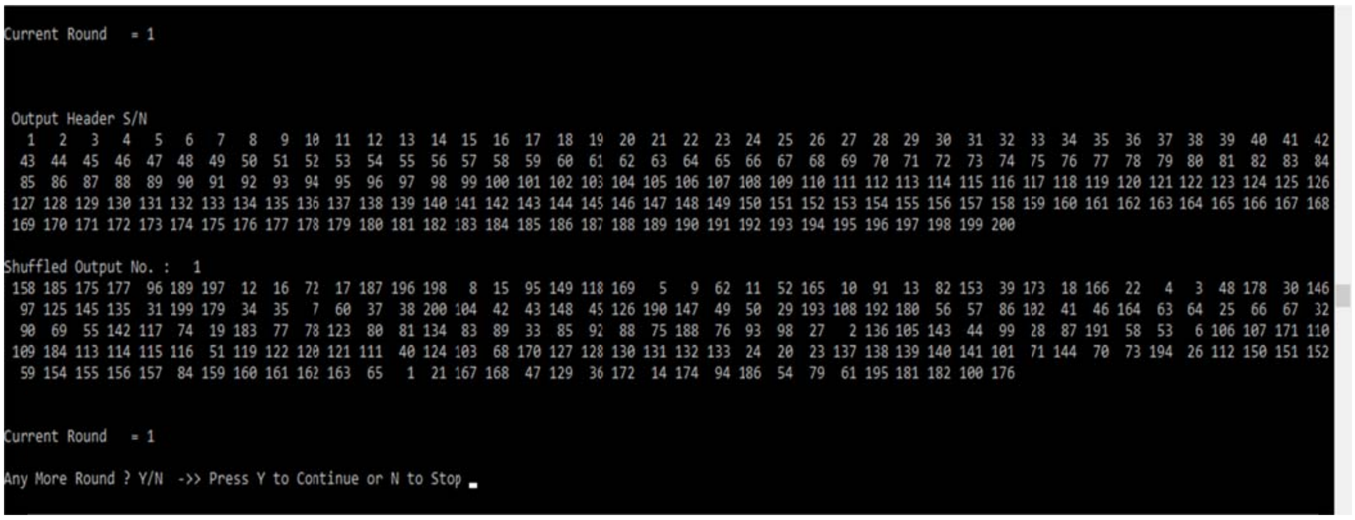
3.2 Results of the randomization mechanism C program for the online collaborative marking system: Case 2 with a class of 200 students all randomized as one row

In Figure 6 is shown the screenshot of the program output for setting the number of students considered in the collaborative marking. In this case, 200 students are expected and they are to be handled in 1 row of 200 students per row, so, KountMx =1. Hence, the needed value on the input line is 200, as shown in Figure 6. The screenshot of the program output for the 200 students generated and stored in the randomization output array is given in Figure 7, while the screenshot of the program output for the validation results of the 200 students generated and stored in the randomization output array is shown in Figure 8. It can be seen from Figure 8 that there is no '0' in the validation output array for the 200 students.



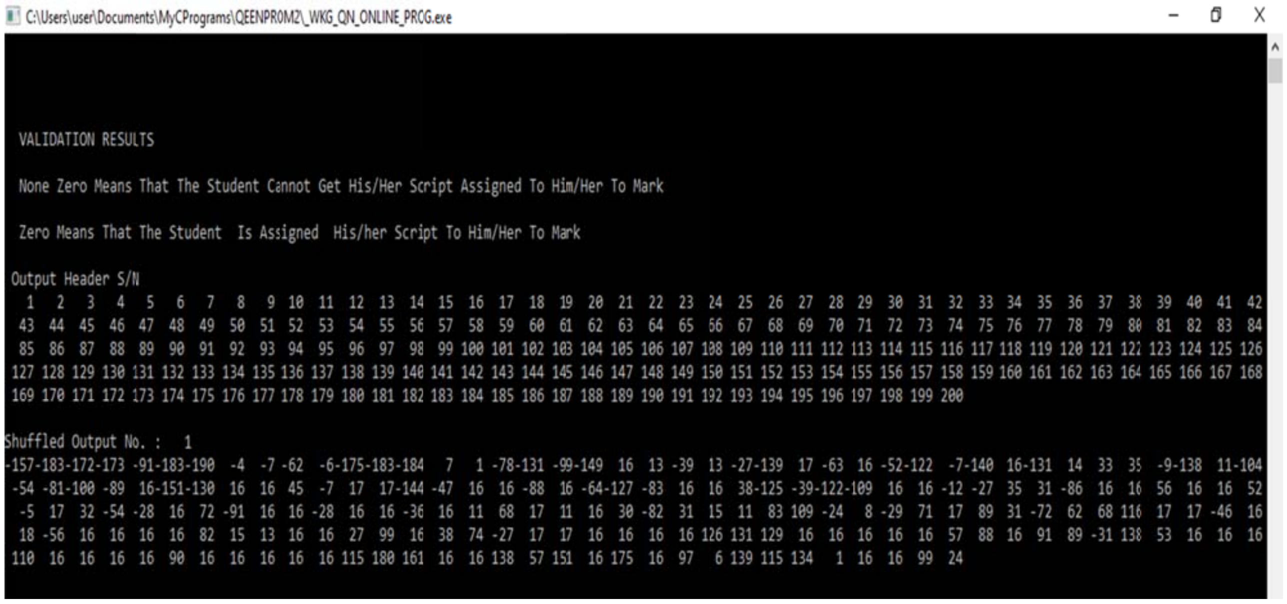
```
C:\Users\user\Documents\MyCPrograms\QEENPROM2\_WKG_QN_ONLINE_PROG.exe


Enter The Number Of Students [The Number Must Be Greater Than 3 ]:   200
```

Figure 6 The screenshot of the program output for setting the number of students considered in the collaborative marking



```
Current Round   = 1


 Output Header S/N
   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42
  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84
  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200

Shuffled Output No. :   1
 158 185 175 177  96 189 197  12  16  72  17 187 196 198   8  15  95 149 118 169   5   9  62  11  52 165  10  91  13  82 153  39 173  18 166  22   4   3  48 178  30 146
  97 125 145 135  31 199 179  34  35   7  60  37  38 200 104  42  43 148  45 126 190 147  49  50  29 193 108 192 180  56  57  86 102  41  46 164  63  64  25  66  67  32
  90  69  55 142 117  74  19 183  77  78 123  80  81 134  83  89  33  85  92  88  75 188  76  93  98  27   2 136 105 143  44  99  28  87 191  58  53   6 106 107 171 110
 109 184 113 114 115 116  51 119 122 120 121 111  40 124 103  68 170 127 128 130 131 132 133  24  20  23 137 138 139 140 141 101  71 144  70  73 194  26 112 150 151 152
  59 154 155 156 157  84 159 160 161 162 163  65   1  21 167 168  47 129  36 172  14 174  94 186  54  79  61 195 181 182 100 176


Current Round   = 1

Any More Round ? Y/N  ->> Press Y to Continue or N to Stop _
```

Figure 7 The screenshot of the program output for the 200 students generated and stored in the randomization output array

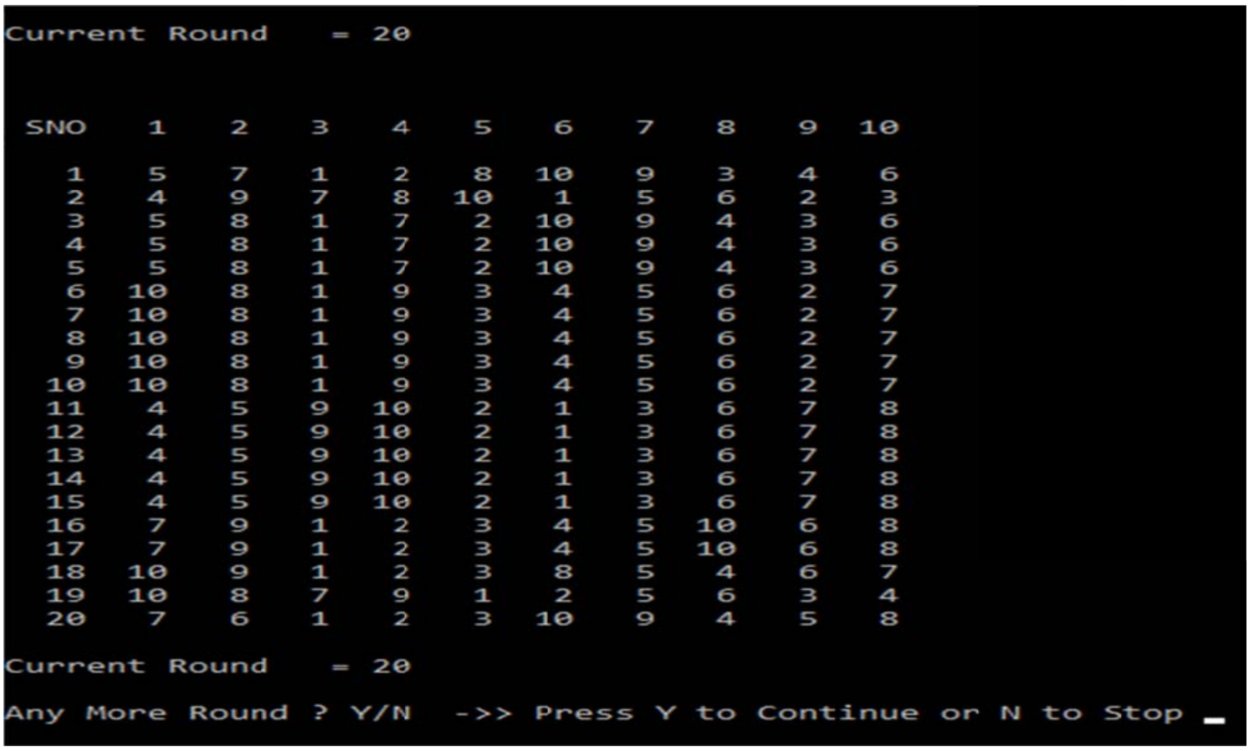


```
C:\Users\user\Documents\MyCPrograms\QEENPROM2\_WKG_QN_ONLINE_PRCG.exe


 VALIDATION RESULTS

 None Zero Means That The Student Cannot Get His/Her Script Assigned To Him/Her To Mark

 Zero Means That The Student  Is Assigned  His/her Script To Him/Her To Mark

 Output Header S/N
   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42
  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84
  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200

Shuffled Output No. :   1
-157-183-172-173 -91-183-190  -4  -7 -62  -6-175-183-184   7   1 -78-131 -99-149  16  13 -39  13 -27-139  17 -63  16 -52-122  -7-140  16-131  14  33  35  -9-138  11-104
 -54 -81-100 -89  16-151-130  16  16  45  -7  17  17-144 -47  16  16 -88  16 -64-127 -83  16  16  38-125 -39-122-109  16  16 -12 -27  35  31 -86  16  16  56  16  16  52
  -5  17  32 -54 -28  16  72 -91  16  16 -28  16  16 -36  16  11  68  17  11  16  30 -82  31  15  11  83 109 -24   8 -29  71  17  89  31 -72  62  68 116  17  17 -46  16
  18 -56  16  16  16  16  82  15  13  16  16  27  99  16  38  74 -27  17  17  16  16  16  16 126 131 129  16  16  16  16  16  57  88  16  91  89 -31 138  53  16  16  16
 110  16  16  16  16  90  16  16  16  16  16 115 180 161  16  16 138  57 151  16 175  16  97   6 139 115 134   1  16  16  99  24
```

Figure 8 The screenshot of the program output for the validation results of the 200 students and stored in the randomization output array

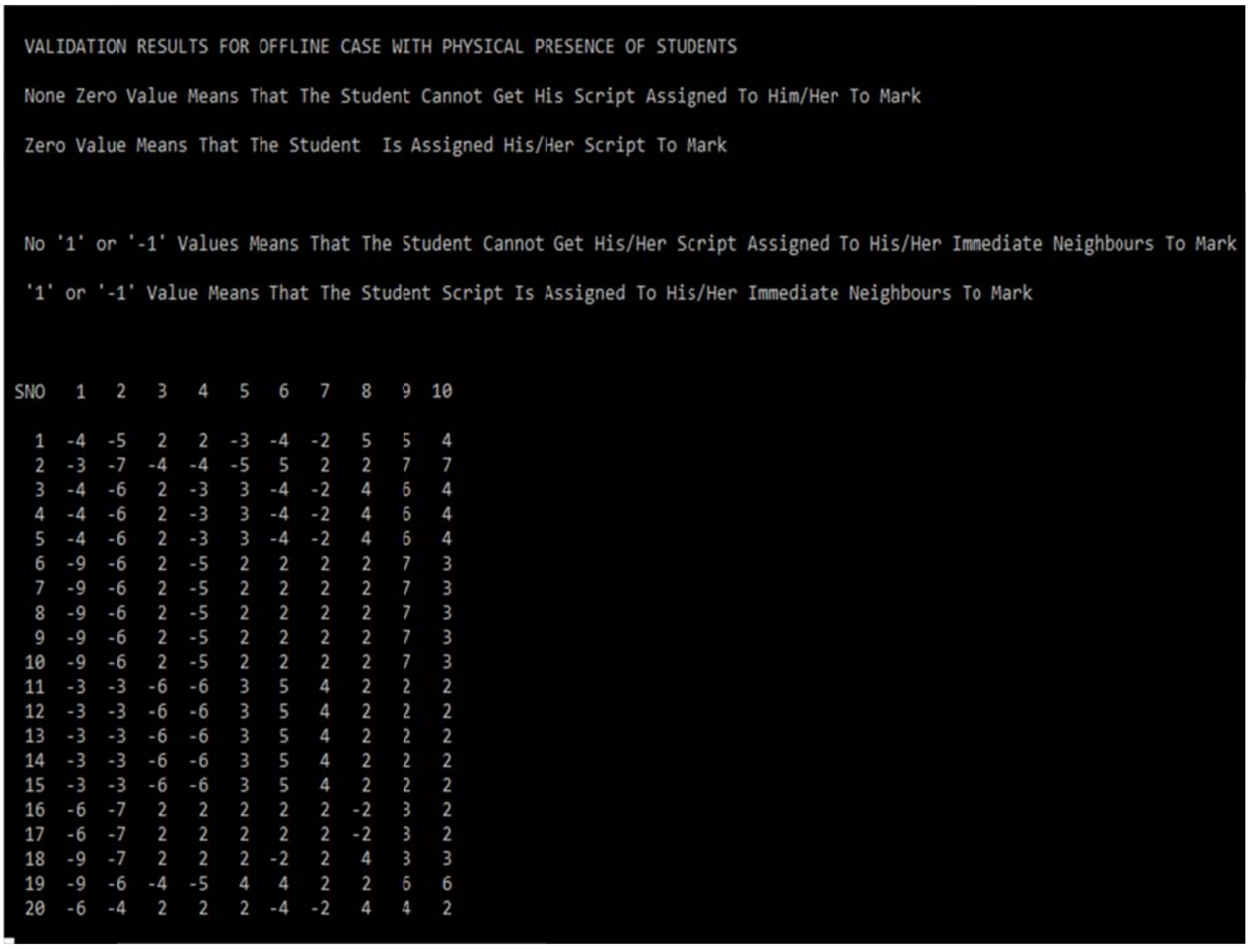
3.3 Results of the randomization mechanism C program for the offline collaborative marking system: Case 1 with a class of 200 students arranged in 20 rows of 10 students per row of 10

The screenshot of the program output for the 20 rows generated and stored in the randomization output array for the offline scenario is given in Figure 9 while the screenshot of the program output for the validation results of the 20 rows generated and stored in the randomization output array is shown in Figure 10. It can be seen from Figure 10 that there is no 1 or -1 and 0 in the validation output array for the 20 rows of 10 per row. Again, the total number in the output of Figure 9 and Figure 10 is 200 which corresponds to the total number of students considered in this case. The results show that in the first row (row with SNO =1), student 1 is assigned the script of student 5, student 2 is assigned the script of student 7 and student 3 is assigned the script of student 1. Similarly, in the 20th row (row with SNO =20), student 1 is assigned the script of student 7, student 2 is assigned the script of student 6 and student 3 is assigned the script of student 1. Accordingly, in Figure 10, there is no ‘1’ or ‘-1’ or ‘0’ in the output which shows that the randomization process is acceptable; no student will be assigned his script to mark and no student will be assigned the script from his or her immediate neighbors.



```
Current Round    = 20


 SNO   1   2   3   4   5   6   7   8   9  10

   1   5   7   1   2   8  10   9   3   4   6
   2   4   9   7   8  10   1   5   6   2   3
   3   5   8   1   7   2  10   9   4   3   6
   4   5   8   1   7   2  10   9   4   3   6
   5   5   8   1   7   2  10   9   4   3   6
   6  10   8   1   9   3   4   5   6   2   7
   7  10   8   1   9   3   4   5   6   2   7
   8  10   8   1   9   3   4   5   6   2   7
   9  10   8   1   9   3   4   5   6   2   7
  10  10   8   1   9   3   4   5   6   2   7
  11   4   5   9  10   2   1   3   6   7   8
  12   4   5   9  10   2   1   3   6   7   8
  13   4   5   9  10   2   1   3   6   7   8
  14   4   5   9  10   2   1   3   6   7   8
  15   4   5   9  10   2   1   3   6   7   8
  16   7   9   1   2   3   4   5  10   6   8
  17   7   9   1   2   3   4   5  10   6   8
  18  10   9   1   2   3   8   5   4   6   7
  19  10   8   7   9   1   2   5   6   3   4
  20   7   6   1   2   3  10   9   4   5   8

Current Round    = 20

Any More Round ? Y/N  ->> Press Y to Continue or N to Stop _
```

Figure 9 The screenshot of the program output for the 20 rows generated and stored in the randomization output array for the offline scenario

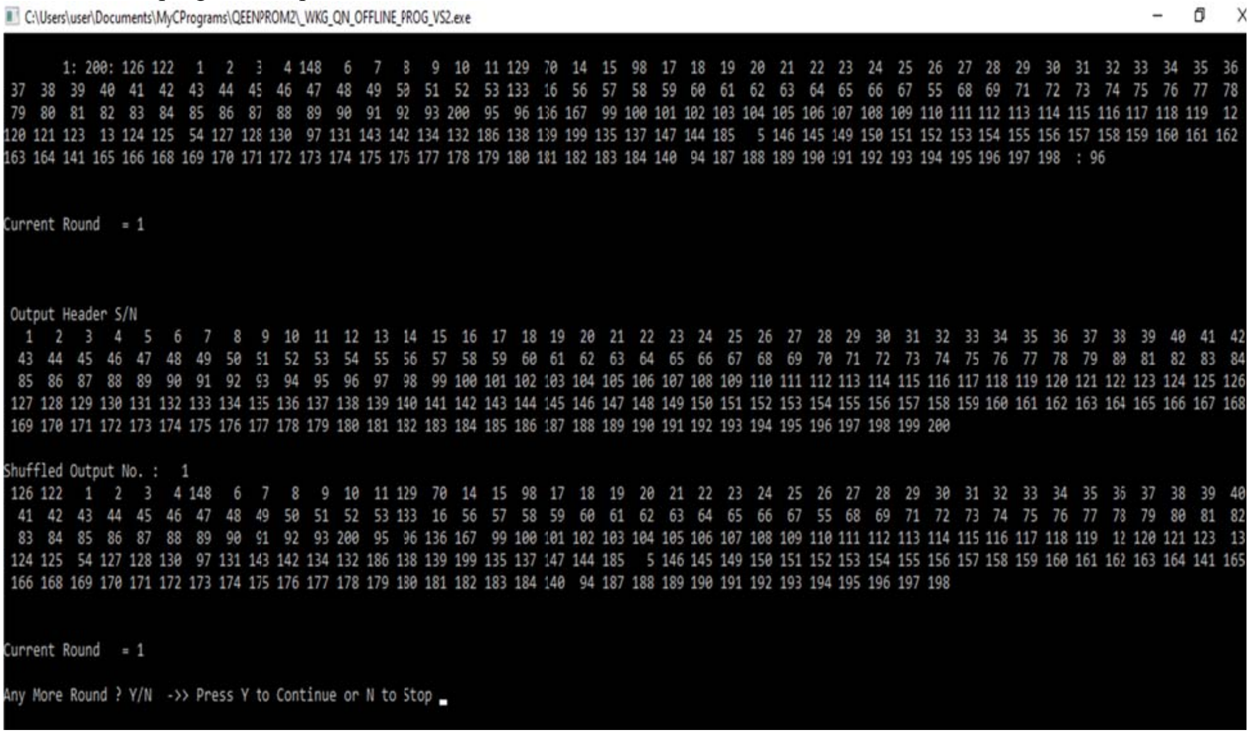


```
 VALIDATION RESULTS FOR OFFLINE CASE WITH PHYSICAL PRESENCE OF STUDENTS

 None Zero Value Means That The Student Cannot Get His Script Assigned To Him/Her To Mark

 Zero Value Means That The Student  Is Assigned His/Her Script To Mark


 No '1' or '-1' Values Means That The Student Cannot Get His/Her Script Assigned To His/Her Immediate Neighbours To Mark

 '1' or '-1' Value Means That The Student Script Is Assigned To His/Her Immediate Neighbours To Mark


SNO   1   2   3   4   5   6   7   8   9  10

  1  -4  -5   2   2  -3  -4  -2   5   5   4
  2  -3  -7  -4  -4  -5   5   2   2   7   7
  3  -4  -6   2  -3   3  -4  -2   4   6   4
  4  -4  -6   2  -3   3  -4  -2   4   6   4
  5  -4  -6   2  -3   3  -4  -2   4   6   4
  6  -9  -6   2  -5   2   2   2   2   7   3
  7  -9  -6   2  -5   2   2   2   2   7   3
  8  -9  -6   2  -5   2   2   2   2   7   3
  9  -9  -6   2  -5   2   2   2   2   7   3
 10  -9  -6   2  -5   2   2   2   2   7   3
 11  -3  -3  -6  -6   3   5   4   2   2   2
 12  -3  -3  -6  -6   3   5   4   2   2   2
 13  -3  -3  -6  -6   3   5   4   2   2   2
 14  -3  -3  -6  -6   3   5   4   2   2   2
 15  -3  -3  -6  -6   3   5   4   2   2   2
 16  -6  -7   2   2   2   2   2  -2   3   2
 17  -6  -7   2   2   2   2   2  -2   3   2
 18  -9  -7   2   2   2  -2   2   4   3   3
 19  -9  -6  -4  -5   4   4   2   2   6   6
 20  -6  -4   2   2   2  -4  -2   4   4   2
```

Figure 10 The screenshot of the program output for the validation results of the 20 rows generated and stored in the randomization output array for the offline scenario

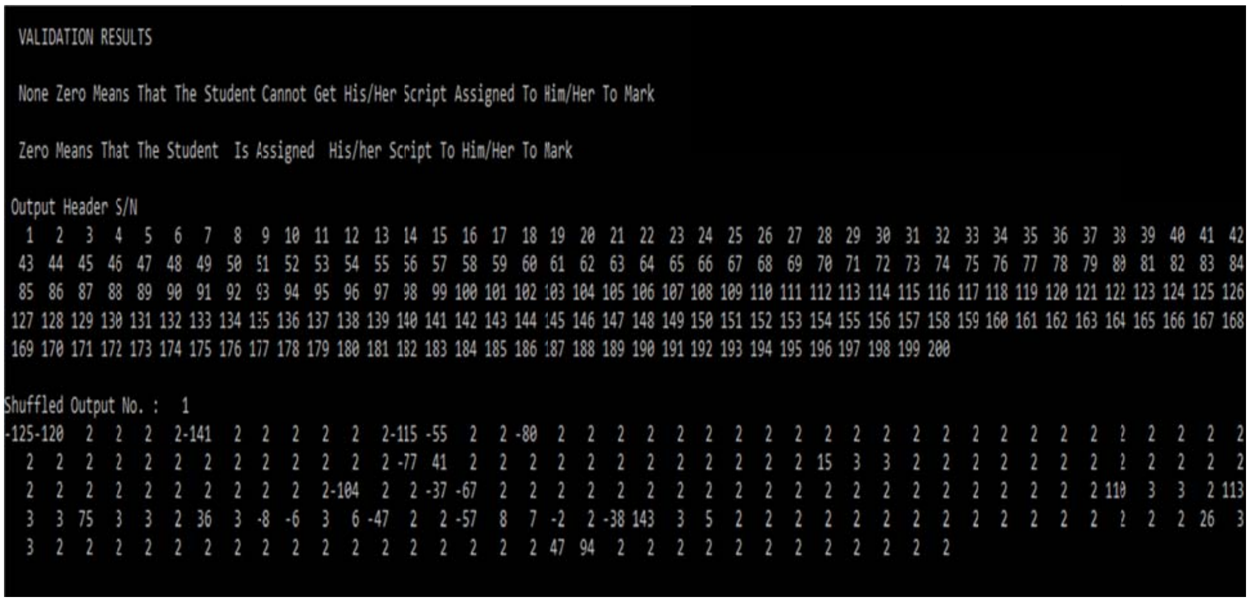
3.4 Results of the randomization mechanism C program for the offline collaborative marking system: Case 2 with a class of 200 students all randomized as one row

In this case, 200 students are expected and they are to be handled in 1 row of 200 students per row, so, KountMx =1. Hence, the needed value on the input line is 200. The screenshot of the program output for the 200 students generated and stored in the randomization output array is given in Figure 11 for the offline scenario, while the screenshot of the program output for the validation results of the 200 students generated and stored in the randomization output array is shown in Figure 12. It can be seen from Figure 12 that there is no 1 or -1 or 0 in the validation output array for the 200 students.



```
C:\Users\user\Documents\MyCPrograms\QEENPROM2\_WKG_QN_OFFLINE_PROG_VS2.exe

      1: 200: 126 122   1   2   3   4 148   6   7   8   9  10  11 129  70  14  15  98  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
 37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53 133  16  56  57  58  59  60  61  62  63  64  65  66  67  55  68  69  71  72  73  74  75  76  77  78
 79  80  81  82  83  84  85  86  87  88  89  90  91  92  93 200  95  96 136 167  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119  12
120 121 123  13 124 125  54 127 128 130  97 131 143 142 134 132 186 138 139 199 135 137 147 144 185   5 146 145 149 150 151 152 153 154 155 156 157 158 159 160 161 162
163 164 141 165 166 168 169 170 171 172 173 174 175 175 177 178 179 180 181 182 183 184 140  94 187 188 189 190 191 192 193 194 195 196 197 198  : 96


Current Round   = 1



 Output Header S/N
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42
 43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84
 85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200

Shuffled Output No. :   1
126 122   1   2   3   4 148   6   7   8   9  10  11 129  70  14  15  98  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
 41  42  43  44  45  46  47  48  49  50  51  52  53 133  16  56  57  58  59  60  61  62  63  64  65  66  67  55  68  69  71  72  73  74  75  76  77  78  79  80  81  82
 83  84  85  86  87  88  89  90  91  92  93 200  95  96 136 167  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119  12 120 121 123  13
124 125  54 127 128 130  97 131 143 142 134 132 186 138 139 199 135 137 147 144 185   5 146 145 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 141 165
166 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 140  94 187 188 189 190 191 192 193 194 195 196 197 198


Current Round   = 1

Any More Round ? Y/N  ->> Press Y to Continue or N to Stop _
```

Figure 11 The screenshot of the program output for the 200 students generated and stored in the randomization output array for the offline scenario



```
 VALIDATION RESULTS

 None Zero Means That The Student Cannot Get His/Her Script Assigned To Him/Her To Mark

 Zero Means That The Student  Is Assigned  His/her Script To Him/Her To Mark

 Output Header S/N
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42
 43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84
 85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200

Shuffled Output No. :   1
-125-120   2   2   2   2-141   2   2   2   2   2   2-115 -55   2   2 -80   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
  2   2   2   2   2   2   2   2   2   2   2   2   2 -77  41   2   2   2   2   2   2   2   2   2   2   2  15   3   3   2   2   2   2   2   2   2   2   2   2   2   2
  2   2   2   2   2   2   2   2   2   2   2-104   2   2 -37 -67   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2 110   3   3   2 113
  3   3  75   3   3   2  36   3  -8  -6   3   6 -47   2   2 -57   8   7  -2   2 -38 143   3   5   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2  26   3
  3   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2  47  94   2   2   2   2   2   2   2   2   2   2   2   2
```

Figure 12 The screenshot of the program output for the validation results of the 200 students and stored in the randomization output array for the offline scenario

4. Conclusion

The development of a program that can be used to shuffle students' answer booklets and assign them to the students for either online or offline collaborative marking system is presented. The program is coded in C programing language. Provisions are made for subdividing the students into row and columns or for shuffling the entire students as one unit or row. In all, the conditions for effective shuffling of the students in the online and offline cases are stipulated and the program shuffling output are validated to ascertain that the conditions are met. In all, the numerical examples presented showed that the program can effectively shuffle the students in both online and offline scenarios.

References

1. Allen, R. L., & Liou, D. D. (2019). Managing whiteness: The call for educational leadership to breach the contractual expectations of white supremacy. *Urban Education*, *54*(5), 677-705.
2. Spiteri, J. (2021). Quality early childhood education for all and the Covid-19 crisis: A viewpoint. *Prospects*, *51*(1), 143-148.
3. Sondel, B., Kretchmar, K., & Hadley Dunn, A. (2019). “Who do these people want teaching their children?” White saviorism, colorblind racism, and anti-blackness in “no excuses” charter schools. *Urban Education*, 0042085919842618.
4. Carter, S., & Abawi, L. A. (2018). Leadership, inclusion, and quality education for all. *Australasian Journal of Special and Inclusive Education*, *42*(1), 49-64.
5. Madani, R. A. (2019). Analysis of Educational Quality, a Goal of Education for All Policy. *Higher Education Studies*, *9*(1), 100-109.
6. Donnor, J. K. (2013). Education as the property of whites: African Americans' continued quest for good schools. In *Handbook of critical race theory in education* (pp. 215-223). Routledge.
7. Hudson, B. (2019). Epistemic quality for equitable access to quality education in school mathematics. *Journal of Curriculum Studies*, *51*(4), 437-456.
8. Didham, R. J., & Ofei-Manu, P. (2018). Advancing policy to achieve quality education for sustainable development. *Issues and trends in Education for Sustainable Development*, 87.
9. Krishnaratne, S., & White, H. (2013). *Quality education for all children? What works in education in developing countries* (No. 0000-0). International Initiative for Impact Evaluation (3ie).
10. Alexander, R. (2008). *Education For All, The Quality Imperative and the Problem of Pedagogy. CREATE Pathways to Access. Research Monograph No. 20*.
11. Pring, R., Hayward, G., Hodgson, A., Johnson, J., Keep, E., Oancea, A., ... & Wilde, S. (2012). *Education for All: The future of education and training for 14-19 year-olds*. Routledge.
12. Rockström, J., Sachs, J. D., Öhman, M. C., & Schmidt-Traub, G. (2013). *Sustainable development and planetary boundaries*. Sustainable Development Solutions Network..
13. Liu, X., Schwaag Serger, S., Tagscherer, U., & Chang, A. Y. (2017). Beyond catch-up—can a new innovation policy help China overcome the middle income trap?. *Science and Public Policy*, *44*(5), 656-669.
14. Malik, R. S. (2018). Educational challenges in 21st century and sustainable development. *Journal of Sustainable Development Education and Research*, *2*(1), 9-20.
15. Medvedev, D. (2015). A new reality: Russia and global challenges. *Russian Journal of Economics*, *1*(2), 109-129.
16. Chimombo, J. P. G. (2005). Quantity versus quality in education: Case studies in Malawi. *International Review of Education*, *51*(2), 155-172.
17. Osborn, D., Cutter, A., & Ullah, F. (2015). Universal sustainable development goals. *Understanding the Transformational Challenge for Developed Countries*.
18. Crespi, G., Fernández-Arias, E., & Stein, E. (2014). Rethinking productive development. In *Rethinking productive development* (pp. 3-31). Palgrave Macmillan, New York.
19. Pucciarelli, F., & Kaplan, A. (2016). Competition and strategy in higher education: Managing complexity and uncertainty. *Business Horizons*, *59*(3), 311-320.
20. Guimón, J. (2013). Promoting university-industry collaboration in developing countries. *World Bank*, *3*, 12-48.
21. Leal Filho, W., Tripathi, S. K., Andrade Guerra, J. B. S. O. D., Giné-Garriga, R., Orlovic Lovren, V., & Willats, J. (2019). Using the sustainable development goals towards a better understanding of sustainability challenges. *International Journal of Sustainable Development & World Ecology*, *26*(2), 179-190.
22. Strimel, G. J., Grubbs, M. E., & Wells, J. G. (2016). Engineering education: A clear decision. *Technology and Engineering Teacher*, *76*(4), 18.
23. Wells, J. G. (2010). Research on Teaching and Learning in Science Education: Potentials of Technology Education. Council on Technology and Engineering Teacher Education.
24. Talanquer, V. (2014). DBER and STEM education reform: Are we up to the challenge?. *Journal of Research in Science Teaching*, *51*(6), 809-819.
25. C. Bullock, E. (2017). Only STEM can save us? Examining race, place, and STEM

education as property. *Educational Studies*, *53*(6), 628-641.

26. Johnson, C. C., Peters-Burton, E. E., & Moore, T. J. (Eds.). (2015). *STEM road map: A framework for integrated STEM education*. Routledge.
27. Xu, M., Williams, P. J., Gu, J., & Zhang, H. (2020). Hotspots and trends of technology education in the International Journal of Technology and Design Education: 2000–2018. *International Journal of Technology and Design Education*, *30*(2), 207-224.
28. Wells, J. G. (2019). STEM education: The potential of technology education. Council on Technology and Engineering Teacher Education.
29. De Vries, M. J. (Ed.). (2018). *Handbook of technology education*. Cham: Springer.
30. Pinelli, T. E., & Haynie III, W. J. (2010). A Case for the Nationwide Inclusion of Engineering in the K-12 Curriculum via Technology Education. *Journal of Technology Education*, *21*(2), 52-68.
31. Daugherty, M. K., Carter, V., & Swagerty, L. (2014). Elementary STEM education: the future for technology and engineering education?. *Journal of STEM teacher education*, *49*(1), 7.
32. Tytler, R. (2020). STEM education for the twenty-first century. *Integrated Approaches to STEM Education*, 21-43.
33. Ejiwale, J. A. (2013). Barriers to successful implementation of STEM education. *Journal of Education and Learning*, *7*(2), 63-74.
34. Marrero, M. E., Gunning, A. M., & Germain-Williams, T. (2014). What is STEM education?. *Global Education Review*, *1*(4).
35. Freeman, B., Marginson, S., & Tytler, R. (2019). An international view of STEM education. In *STEM Education 2.0* (pp. 350-363). Brill.
36. Belás, J., Kmecová, I., & Čepel, M. (2020). Availability of human capital and the development of the public infrastructure in the context of business activities of SMEs. *Administratie si Management Public*.
37. Avdeeva, E., Davydova, T., Skripnikova, N., & Kochetova, L. (2019). Human resource development in the implementation of the concept of “smart cities”. In *E3S Web of Conferences* (Vol. 110, p. 02139). EDP Sciences.
38. Anderson, W., & Sanga, J. J. (2019). Academia–industry partnerships for hospitality and tourism education in Tanzania. *Journal of Hospitality & Tourism Education*, *31*(1), 34-48.
39. Asamani, J. A., Chebere, M. M., Barton, P. M., D’Almeida, S. A., Odame, E. A., & Oppong, R. (2018). Forecast of healthcare facilities and health workforce requirements for the public sector in Ghana, 2016–2026. *International journal of health policy and management*, *7*(11), 1040.
40. Salam, S., Zeng, J., Pathan, Z. H., Latif, Z., & Shaheen, A. (2018). Impediments to the integration of ICT in public schools of contemporary societies: A review of literature. *Journal of Information Processing Systems*, *14*(1), 252-269.
41. Thi Hang, N., Thi Tinh, D., Ngoc Huy, D. T., & Hong Nhung, P. T. (2021). Educating and training labor force Under Covid 19; impacts to meet market demand in Vietnam during globalization and integration era.
42. Cabral, C., & Dhar, R. L. (2019). Skill development research in India: a systematic literature review and future research agenda. *Benchmarking: An International Journal*.
43. Eze, S. C., Chinedu-Eze, V. C., & Bello, A. O. (2018). The utilisation of e-learning facilities in the educational delivery system of Nigeria: a study of M-University. *International Journal of Educational Technology in Higher Education*, *15*(1), 1-20.
44. Qasem, Y. A., Abdullah, R., Jusoh, Y. Y., Atan, R., & Asadi, S. (2019). Cloud computing adoption in higher education institutions: A systematic review. *IEEE Access*, *7*, 63722-63744.
45. Benavides, L. M. C., Tamayo Arias, J. A., Arango Serna, M. D., Branch Bedoya, J. W., & Burgos, D. (2020). Digital transformation in higher education institutions: A systematic literature review. *Sensors*, *20*(11), 3291.
46. Lassoued, Z., Alhendawi, M., & Bashitialshaaer, R. (2020). An exploratory study of the obstacles for achieving quality in distance learning during the COVID-19 pandemic. *Education sciences*, *10*(9), 232.
47. Panda, S. (2018). Constraints faced by women entrepreneurs in developing countries: review and ranking. *Gender in Management: An International Journal*.
48. Etomes, S. E., & Lyonga, F. I. N. (2020). STUDENT-TEACHER RATIO AND STUDENTS’ACADEMIC PERFORMANCE IN PUBLIC UNIVERSITIES: THE CASE OF THE UNIVERSITY OF BUEA, CAMEROON. *European Journal of Education Studies*, *7*(6).
49. Redding, C. (2019). A teacher like me: A review of the effect of student–teacher

racial/ethnic matching on teacher perceptions of students and student academic and behavioral outcomes. *Review of educational research*, *89*(4), 499-535.

50. Valkov, P., & Lavrentsova, E. (2019). The Effects of Student-Teacher and Student Student Relationship on School Engagement: An Empirical Research in Bulgaria. *Pedagogy*, *91*(3), 320-331.
51. Joshi, E., Doan, S., & Springer, M. G. (2018). Student-teacher race congruence: New evidence and insight from Tennessee. *AERA Open*, *4*(4), 2332858418817528.
52. Longobardi, C., Settanni, M., Lin, S., & Fabris, M. A. (2021). Student–teacher relationship quality and prosocial behaviour: The mediating role of academic achievement and a positive attitude towards school. *British Journal of Educational Psychology*, *91*(2), 547-562.
53. Apriesnig, J. L., Manning, D. T., Suter, J. F., Magzamen, S., & Cross, J. E. (2020). Academic stars and Energy Stars, an assessment of student academic achievement and school building energy efficiency. *Energy Policy*, *147*, 111859.
54. Splett, J. W., Smith-Millman, M., Raborn, A., Brann, K. L., Flaspohler, P. D., & Maras, M. A. (2018). Student, teacher, and classroom predictors of between-teacher variance of students' teacher-rated behavior. *School Psychology Quarterly*, *33*(3), 460.
55. Leung, W. T. V., Tam, T. Y. T., Pan, W. C., Wu, C. D., Lung, S. C. C., & Spengler, J. D. (2019). How is environmental greenness related to students' academic performance in English and Mathematics?. *Landscape and Urban Planning*, *181*, 118-124.
56. Gezim, B. A. R. A., & Xhomara, N. (2020). The effect of student-centered teaching and problem-based learning on academic achievement in science. *Journal of Turkish Science Education*, *17*(2), 180-199.
57. Xu, D., & Li, Q. (2018). Gender achievement gaps among Chinese middle school students and the role of teachers' gender. *Economics of Education Review*, *67*, 82-93.
58. Jadoon, A. I., Khan, F., Syeda Tehmina Naz Bukhari, N. T. S., Gilani, S. Z., Ishfaq, U., & Ullah, M. (2022). Effect of Teacher-Student Relationship on Pro-Social Behavior and Academic Achievement of Secondary School Students. *Indian Journal of Economics and Business*, *21*(1), 331-337.
59. Shirrell, M., Bristol, T. J., & Britton, T. A. (2021). The Effects of Student-Teacher Ethnoracial Matching on Exclusionary Discipline for Asian American, Black, and Latinx Students: Evidence from New York City. EdWorkingPaper No. 21-475. *Annenberg Institute for School Reform at Brown University*.