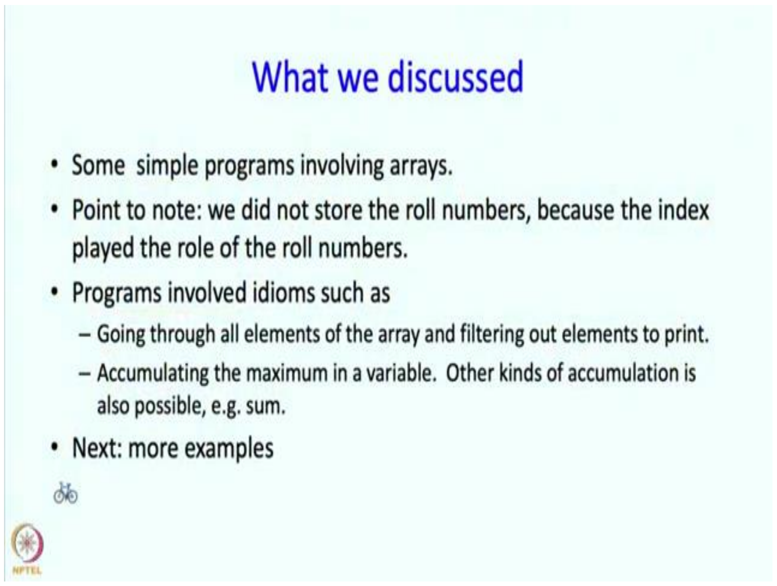
An Introduction to Programming through C++
Professor Abhiram G. Ranade
Department of Computer Science and Engineering
Indian Institute of Technology Bombay
Lecture No. 15 Part - 3
Array Part – 1
Histogram computation

(Refer Slide Time: 0:20)

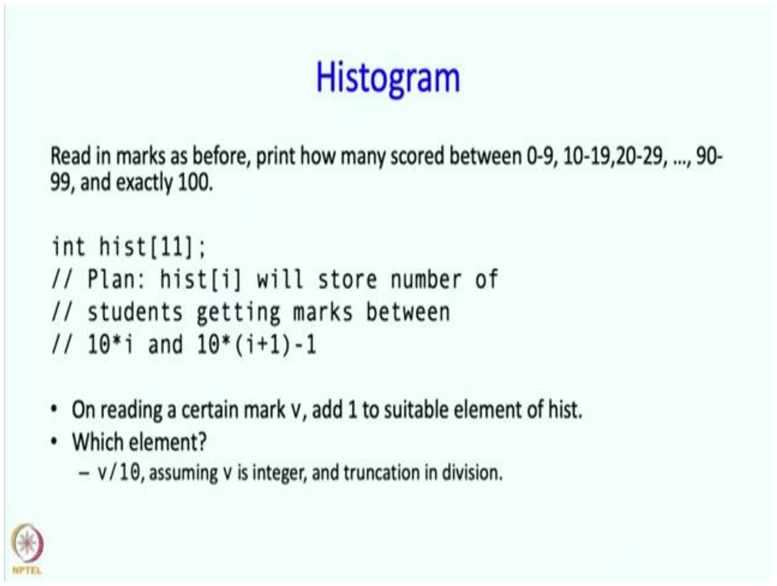


What we discussed

- Some simple programs involving arrays.
- Point to note: we did not store the roll numbers, because the index played the role of the roll numbers.
- Programs involved idioms such as
  - Going through all elements of the array and filtering out elements to print.
  - Accumulating the maximum in a variable. Other kinds of accumulation is also possible, e.g. sum.
- Next: more examples

Hello and welcome back. In the previous segment we saw a few simple programs involving arrays and we discussed some idioms such as filtering and accumulating and we are going to do more examples.

(Refer Slide Time: 0:31)

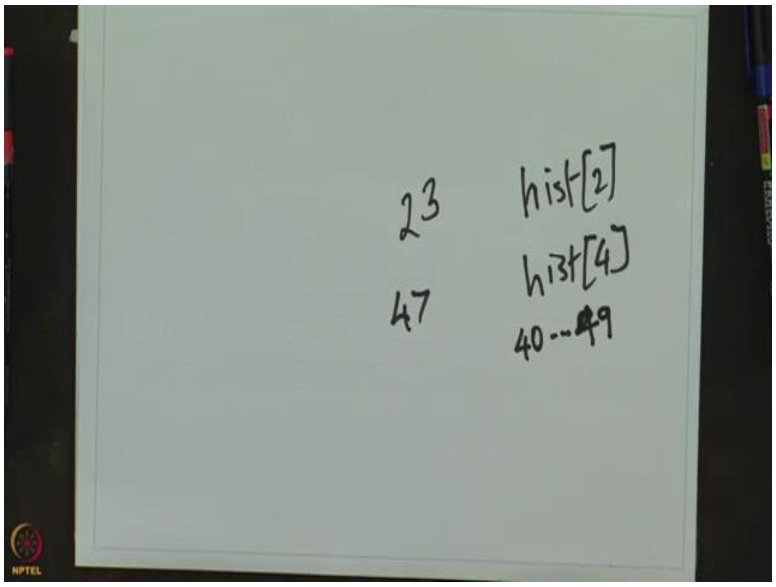


Histogram

Read in marks as before, print how many scored between 0-9, 10-19,20-29, ..., 90-99, and exactly 100.

```
int hist[11];
// Plan: hist[i] will store number of
// students getting marks between
// 10*i and 10*(i+1)-1
```

- On reading a certain mark v, add 1 to suitable element of hist.
- Which element?
  - v/10, assuming v is integer, and truncation in division.



So this example that we are going to do is sort of a rather interesting example, it shows some clever ways in which the array index can be used. So, what is the problem? The problem is that we are supposed to read in marks as before, but we should print how many students scored between 0 and 9, print how many students scored between 10 and 19, how many between 20 and 29, and so on.

This is often done by statisticians to get a general sense of how well the people have scored, so rather that looking at every mark which is lots of numbers to look at, they break it up into ranges and see how many marks fall into the different ranges. So in this case now, our output is going to be 11 numbers, so one number for the range 0 through 99, another number for the range 10 through 19, 20 through 19, all the way to 90 through 19 and also for the range starting at 100.

Of course there is no range, but since a person can get 100 marks, we need to have a way to say that some person has got 100. So there are 11, there are 11 buckets so to say including the last rather small bucket or the last bucket which may think of it, think of it as being between 100 and 109, but of course nobody is going to get more marks than 100 let us say. So we have 11 possible numbers to output or 11 buckets sort to say into which the marks are supposed to be put, so we declare an array called hist[length 11.

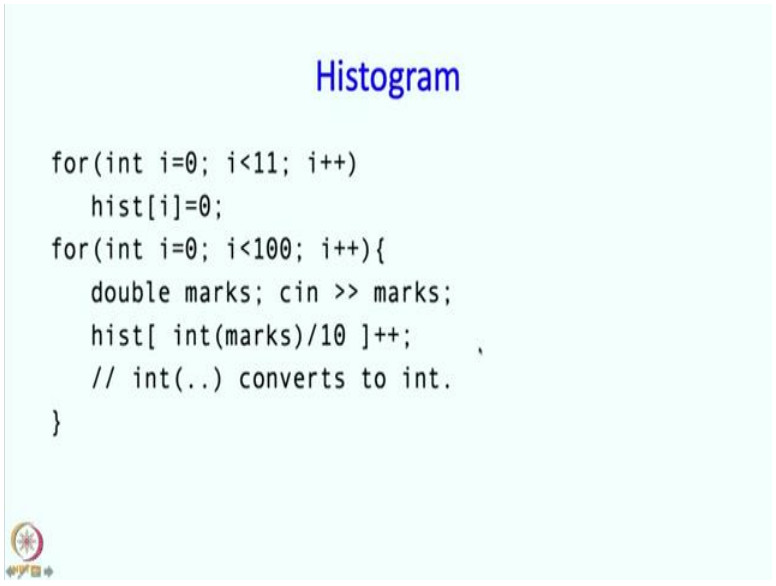
And hist[i] is going to store number of students getting marks between 10i and 10(i+1)-1. So, if I use i equals 0, then this becomes 0 and 10 minus 1, and 9. So that is indeed the first range. If I make this say i equals 5, then it becomes 50 and 60 minus 1 or 59. So those are exactly the ranges that we want and those are the buckets that we want as they are often called.

So, what should we be doing? So, if I read a certain mark from the keyboard, the user is going to type the marks from the keyboard, maybe the teacher is going to, or somebody is going to. Then we have to decide which of these ranges it belongs to and we are going to add 1 to hist of that i. So that is what we are supposed to do. So given a mark v, which of these elements should we increment?

So let us take an example, suppose I am given the mark 58, so what should I do? So I should increment the bucket corresponding to the range 50 through 59, which is the bucket with index 5. So we have chosen these indices conveniently, so in fact the most significant digit of your mark or rather the digit, the number obtained by dropping the least significant digits gives you the index of the element of hist that you are supposed to increment.

So for example, if I get 23 then I should increment hist[2], if I get 47 I should increment hist[4], because hist[4] then this 47 is going to lie in the range 40 through 49 and that is and how many marks lie in this range is exactly what is supposed to be stored in hist[4]. So, if I get a new mark in this range, then this number should go up. So, how do we do this? Given a number in the range 40 through 49, how do we get a 4 out of it? Well, that is very easy.

(Refer Slide Time: 5:11)



Histogram

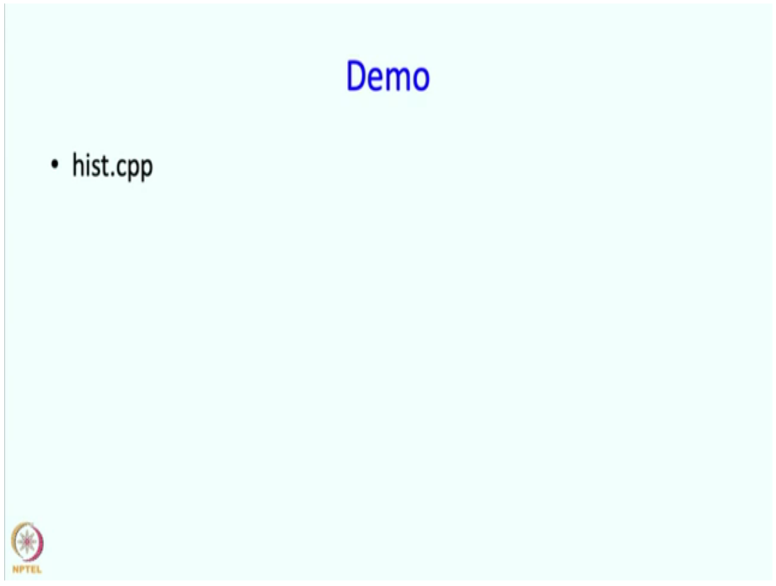
```
for(int i=0; i<11; i++)
   hist[i]=0;
for(int i=0; i<100; i++){
   double marks; cin >> marks;
   hist[ int(marks)/10 ]++;
   // int(..) converts to int.
}
```

So we just do integer division, so we assume that v is an, or we convert v into an integer if it is not already an integer and we divide by 10 and we know that if we divide a integer by another integer, we get the truncated division. So, if we indeed divide say 47 by 10, then we will get 4 because the remainder 7 will be thrown out. So that is all there is to hist.

So our program is that we start with, so we have 11 buckets first of all, so we are going to make all those buckets be 0. So at this point we have not read any marks and therefore, the ranges, all the ranges (containing) contain 0. So no mark has been read. Now, here we are going to read one mark after another and when we read a mark, so we are supposed to figure out, which bucket we should increment and increment it.

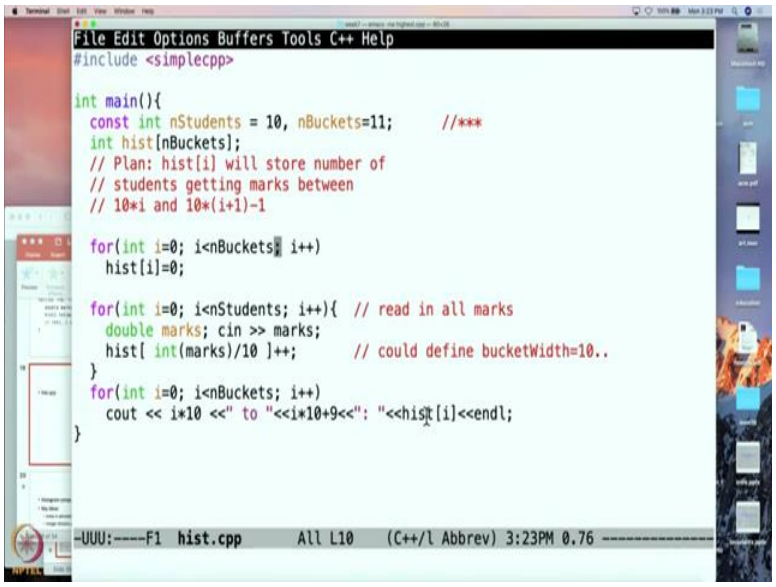
So as we said we should divide, we should divide the marks by 10, but however, notice that our marks are double. So of course our marks could be 47.5 for all you know. So, if I write int of marks then I get first of all, I get the integer part and then I get the integer division, so I get now by doing the integer division I get to know which element of hist I have to increment and so I just go ahead and increment that element, that is all there is to it. So after this I can print out the values.

(Refer Slide Time: 6:48)



Demo

- hist.cpp



```
File Edit Options Buffers Tools C++ Help
#include <simplecpp>

int main(){
  const int nStudents = 10, nBuckets=11;      //***
  int hist[nBuckets];
  // Plan: hist[i] will store number of
  // students getting marks between
  // 10*i and 10*(i+1)-1

  for(int i=0; i<nBuckets; i++)
    hist[i]=0;

  for(int i=0; i<nStudents; i++){  // read in all marks
    double marks; cin >> marks;
    hist[ int(marks)/10 ]++;       // could define bucketWidth=10..
  }
  for(int i=0; i<nBuckets; i++)
    cout << i*10 <<" to "<<i*10+9<<": "<<hist[i]<<endl;
}

-UUU:----F1  hist.cpp      All L10    (C++/l Abbrev) 3:23PM 0.76 -------------
```

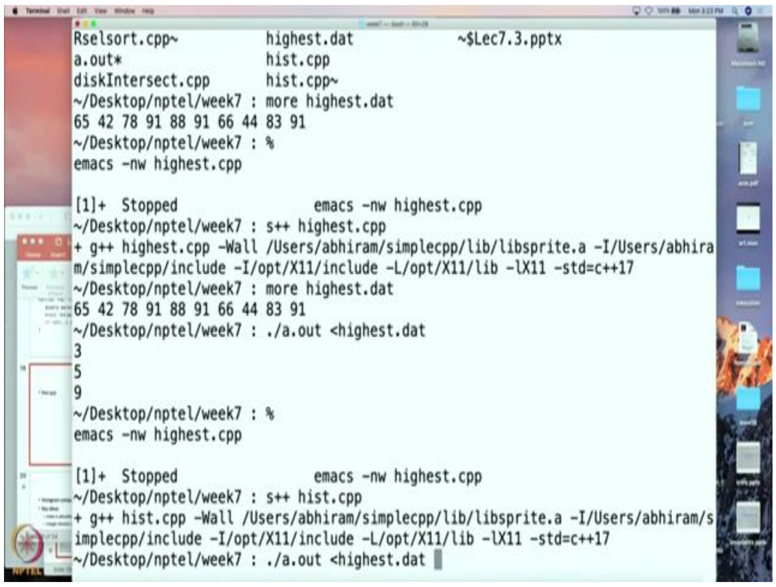
So let us see a quick demo. So this file I have made a little bit different from what you have seen. So first of all instead of putting in numbers such as 10 and 11, which are confusing to the reader, it is much more customary to give these numbers names. So again instead of using 100 students we are going to use 10 students because you do not really want to worry about 100 students in a simple example.

So n students is going to represent how many students there are and therefore, in the rest of the code if you look at nStudents, you will know, okay that this number is actually representing the number of students. Then also there are nbuckets, so nBuckets is a variable which is going to be 11 because we wanted 11 buckets. So we declare an array with 11 buckets and this is the plan that we already had, so there is, so nothing has really changed.

So then we set all the histograms to be initially 0, all the buckets to have 0 elements initially and then we are going to read in all the marks and this time instead of saying 100 over here or 10 over here or whatever it is, we are going to say nStudents. Because we just want to alert the reader that look this is being done once for every student.

Then we read in the marks and then we divide it by 10 but only after converting it to an integer. So I could have, this 10 I could have written as what is the width of a bucket and all of that, I really should do it, but I am being a little bit sloppy here. Alright, and then finally I am going to print out the contents of all the buckets. So I am going to print a message saying in the range i times 10 to i times 10 plus 9 there are so many, so many marks, so that is what it is. So let us do this, let us run it.

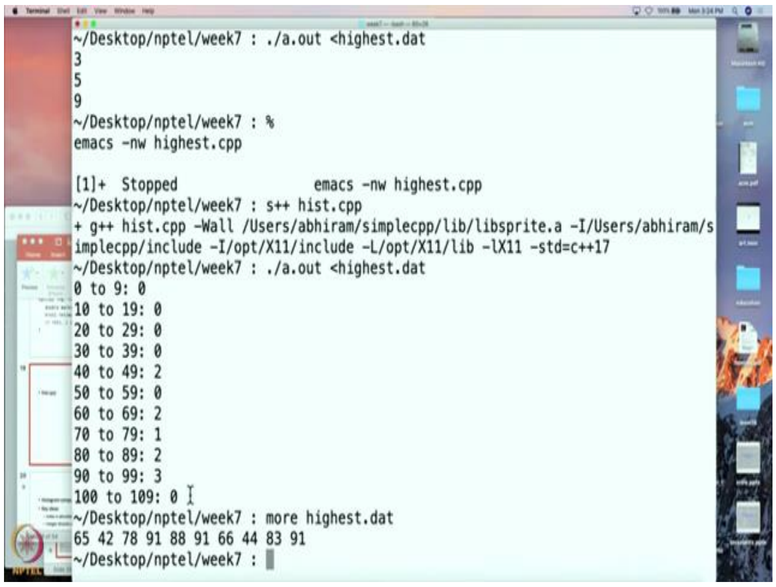
(Refer Slide Time: 8:54)



```
Rselsort.cpp~           highest.dat             ~$Lec7.3.pptx
a.out*                  hist.cpp
diskIntersect.cpp       hist.cpp~
~/Desktop/nptel/week7 : more highest.dat
65 42 78 91 88 91 66 44 83 91
~/Desktop/nptel/week7 : %
emacs -nw highest.cpp

[1]+  Stopped                 emacs -nw highest.cpp
~/Desktop/nptel/week7 : s++ highest.cpp
+ g++ highest.cpp -Wall /Users/abhiram/simplecpp/lib/libsprite.a -I/Users/abhira
m/simplecpp/include -I/opt/X11/include -L/opt/X11/lib -lX11 -std=c++17
~/Desktop/nptel/week7 : more highest.dat
65 42 78 91 88 91 66 44 83 91
~/Desktop/nptel/week7 : ./a.out <highest.dat
3
5
9
~/Desktop/nptel/week7 : %
emacs -nw highest.cpp

[1]+  Stopped                 emacs -nw highest.cpp
~/Desktop/nptel/week7 : s++ hist.cpp
+ g++ hist.cpp -Wall /Users/abhiram/simplecpp/lib/libsprite.a -I/Users/abhiram/s
implecpp/include -I/opt/X11/include -L/opt/X11/lib -lX11 -std=c++17
~/Desktop/nptel/week7 : ./a.out <highest.dat
```



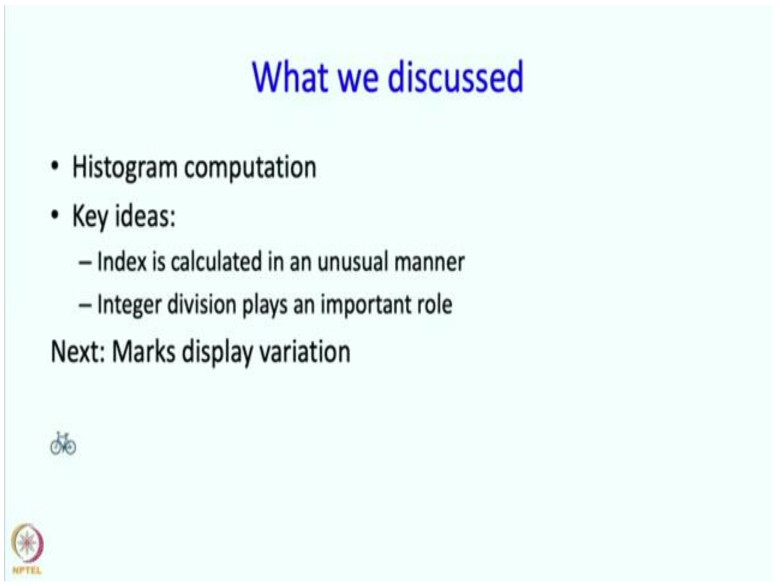
```
~/Desktop/nptel/week7 : ./a.out <highest.dat
3
5
9
~/Desktop/nptel/week7 : %
emacs -nw highest.cpp

[1]+  Stopped                 emacs -nw highest.cpp
~/Desktop/nptel/week7 : s++ hist.cpp
+ g++ hist.cpp -Wall /Users/abhiram/simplecpp/lib/libsprite.a -I/Users/abhiram/s
implecpp/include -I/opt/X11/include -L/opt/X11/lib -lX11 -std=c++17
~/Desktop/nptel/week7 : ./a.out <highest.dat
0 to 9: 0
10 to 19: 0
20 to 29: 0
30 to 39: 0
40 to 49: 2
50 to 59: 0
60 to 69: 2
70 to 79: 1
80 to 89: 2
90 to 99: 3
100 to 109: 0
~/Desktop/nptel/week7 : more highest.dat
65 42 78 91 88 91 66 44 83 91
~/Desktop/nptel/week7 :
```

Let us first compile it, so s++ hist and then, I am going to use the same sort of numbers as I had earlier. So let me show you the numbers, so let me do this, highest.dat, so these are the numbers. And for these numbers this was the histogram, so let us read that, let us check if it is correct. So the histogram that we computed says that there are no numbers in the range 0 through 9.

Indeed there are no such numbers, nothing in fact all the way till 40 to 49 and there are 2 in the range 40 to 49, is that true? Let us see, there is 42 and yes, there is only 44. So there are 2 here, nothing in the range 50 through 59, 2 in the range 60 through 69, well there is 65 and 66, yes that is correct, 70 to 79 there is 1 so there is 78, 80 through 89 there is 88 and 83 and 90 to 99 there are 3, 91, 91 and 91 and nobody gets marks in the range 100 to 109, so it has indeed done the right thing.

(Refer Slide Time: 10:21)



What we discussed

- Histogram computation
- Key ideas:
  - Index is calculated in an unusual manner
  - Integer division plays an important role

Next: Marks display variation

Alright, so what have we discussed? So we have discussed this idea of histogram computation, which I said is an important statistical operation. So by looking at the histogram we can get a rough sense of how people have done. So for example, in our histogram there were no students who were getting marks in the range 0 through 10, 10 to 19, 20 through 29 and all those 0 said that look most students has sort of got good marks. So you can get sort of a rough estimate, so that is what statisticians often like to do.

And the key ideas are that the index for accessing the array element is calculated in an unusual manner. And integer division plays an important role in this. Next we are going to go along with this theme of marks and displaying marks and we are going to look at another variation of that problem. So we will take a quick break.