ROOT
A data analysis framework

Week 7 (05/03/2020)

Outline:

- **Introduction:**
  - **What is ROOT?**
  - **What are the functionalities?**

- **How to use it**
  - **Installation**
  - **How to run your own C++ code**

- **Histogram and Graph**
  - **How to draw a Histogram (1-dim, 2-dim,.. n-dim)**
  - **How to draw a Graph**

- **Input/Output (I/O)**
  - **File handling**
  - **Saving figures**
  - **etc.**

Introduction

https://root.cern.ch

- **Difficulties in big data analysis, limited usages/inefficiency of then available programs like Fortran (1994-1995).**
- **Developed by two Particle Physicists at CERN in 1995:**
  Rene Brun and Fons Rademakers

What is root?

- **A Object Oriented data analysis framework.**
- **Written in C++ (also support interface for Python and Ruby)**
- **Can be used both in interactive and compilation mode (CINT and Cling supported).**

What are functionalities?

- **Histograming, Graph, Curve fitting, Mathematical evaluation (differentiation, integration) matrix and four vector (Lorentz) manipulation, and more.**
- **Multivariate analysis (neural network), Grid computing**
- **3D visualisation (Geometry), Graphics and many more…**

Introduction

Two key features:

Object Oriented:

- **Provides all the advantages what OOPs have (e.g. C++).**
- **Polymorphism, encapsulation and inheritance.**
- **Object may come and go, but basic structure remains for reuse of design.**

Framework based:

- **Less code to write, use and reuse some preexisting basic structure according to your necessity.**
- **Developed mostly by HEP community.**
- **Tested and integrated with rest of the framework-> more reliable.**

- **Used by all CERN based HEP experiments and all major EHEP community around the globe.**
- **Used by Medical and Financial industries.**
- **Community based support (discussion forum: https://root-forum.cern.ch)**
- **Its free!**

An amazing data analysis framework!

How to use it

Download and installation

- **For downloading and installation: https://root.cern.ch/building-root**
- **Check carefully for the *perquisites* depending on your computing environment (Unix, OS X, Window).**
- **Follow the instructions for installation.**
- **After successful instalation, in your terminal:**
- **$root ↲**

```
$ root
   ------------------------------------------------------------------
  | Welcome to ROOT 6.16/00                        https://root.cern |
  |                                (c) 1995-2018, The ROOT Team      |
  | Built for macosx64 on May 03 2019, 21:40:00                      |
  | From tag , 23 January 2019                                       |
  | Try '.help', '.demo', '.license', '.credits', '.quit'/'.q'       |
   ------------------------------------------------------------------

root [0]
```

The interactive session starts from here

Get, set, go…. Enjoy ROOTing

How to use it

Interactive mode

```
Nirbhays-MBP:~ nirbhay$ root
   ------------------------------------------------------
  | Welcome to ROOT 6.16/00                    https://root.cern |
  |                                (c) 1995-2018, The ROOT Team |
  | Built for macosx64 on May 03 2019, 21:40:00                 |
  | From tag , 23 January 2019                                  |
  | Try '.help', '.demo', '.license', '.credits', '.quit'/'.q' |
   ------------------------------------------------------

root [0] cout <<"Hello, this is my root session " << endl;
Hello, this is my root session
root [1] float a = 5.8; float b = 12.4; float c = a + b; float d = a - b;
root [2] cout<<"Addition result= " << c <<"  and substraction= " << d <<
endl;
Addition result= 18.2  and substraction= -6.6
root [3]
```

OR

Open a file: MyTestCode.C

```
{
  float a = 5.8;
  float b = 12.4;
  float c = a + b;
  float d = a - b;
  cout<<"Addition result= " << c <<"  and substraction= " << d << endl;
}
```

```
Terminal — -bash — 82×26
Nirbhays-MBP:~ nirbhay$ root -l MyTestCode.C
root [0]
Processing MyTestCode.C...
Addition result= 18.2  and substraction= -6.6
root [1] .q
Nirbhays-MBP:~ nirbhay$ root -l
root [0] .x MyTestCode.C
Addition result= 18.2  and substraction= -6.6
root [1] .q
Nirbhays-MBP:~ nirbhay$
```

OR Next slide..

How to use it

Compilation mode

Open a file: MyTestCode.C

```
#include <iostream>
using namespace std;

float AddFunction( float x, float y){
  return x + y;

}
float SubstractFunction( float p, float q){
  return p - q;

}

//My main function-----
void MyTestCode(float myA=0., float myB=0.){

  float a = myA;
  float b = myB;
  float c = AddFunction( a, b);
  float d = SubstractFunction( a, b );

  cout<<"Addition result= " << c <<"  and substraction= " << d << endl;

}
```

For compilation, in our terminal:

$root -l ← Opens root without the splash/message screen

root[0].L YourCodeName.C+

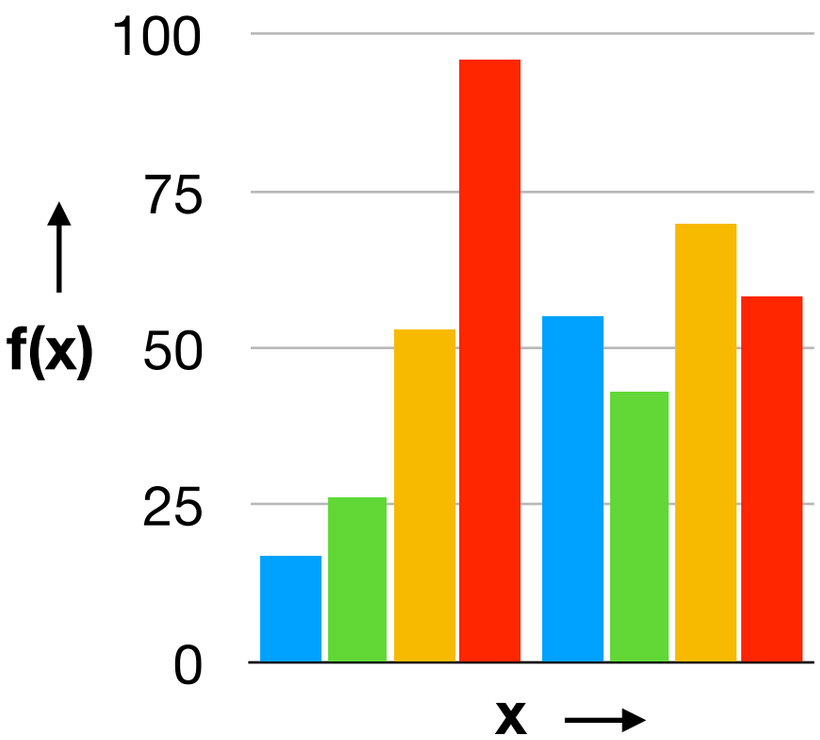
root[1]YourCodeName(your input)

This will create a shared library (.so)

```
Nirbhays-MBP:~ nirbhay$ root -l
root [0] .L MyTestCode.C+
Info in <TMacOSXSystem::ACLiC>: creating shared library /Users/nirbhay/./MyTestCod
e_C.so
root [1] MyTestCode(5.8,12.4)
Addition result= 18.2  and substraction= -6.6
root [2] MyTestCode(10., 6.0)
Addition result= 16  and substraction= 4
root [3] .q
Nirbhays-MBP:~ nirbhay$
```

And many more ….

Histogram



- **Plays an important role for any statistical measurement for scientific interpretation.**
- **Histogram can be 1-dim, 2-dim, 3-dim or n-dimensional.**
- **Some commonly used tools: e.g. GNU plot.**

- **ROOT provides an extensive platform for histogram plotting with wide variety of functionalities.**
- **Interactive editing can be done by root supported Graphical User Interface (GUI).**

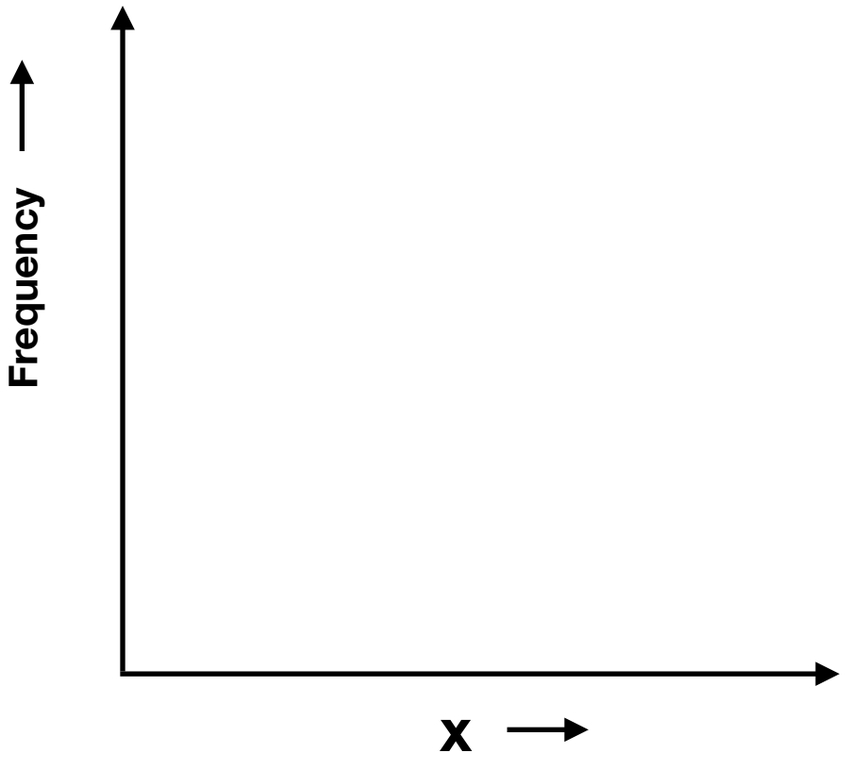
- **Histogram are drawn and edited using the TH1 base class of ROOT**
- **Different types of derived classes for different type of data type**

Most commonly used histogram types

- ✦ **TH1I: integer type, max bin content 2147483647**
- ✦ **TH1F: float type, maximum precision 7 digit**
- ✦ **TH1D: double type, max. precision 14 digit**

Similarly for 2D and 3D, e.g. TH2F, TH3D

Histogram



Marks secured by 200 students in an exam

Sl no.	Marks Secured	Sl no.	Marks Secured
0	59	31	78
1	64	32	76
2	52	33	79
3	71	34	75
4	74	35	60
5	51	36	78
6	10	37	42
7	79	38	89
8	63	39	48
9	42	40	50
10	68	41	63
11	44	42	72
12	79	43	75
13	69	44	79
14	45	45	83
15	58	46	67
16	81	47	71
17	50	48	66
18	36	49	36
19	77	50	66
20	63	51	41
21	65	52	68
22	69	53	43
23	57	54	74
24	85	55	54
25	57	56	58
26	45	57	68
27	93	58	63
28	74	59	72
29	77	….	…
30	99		

Marks are written in a data file named “marks.dat”

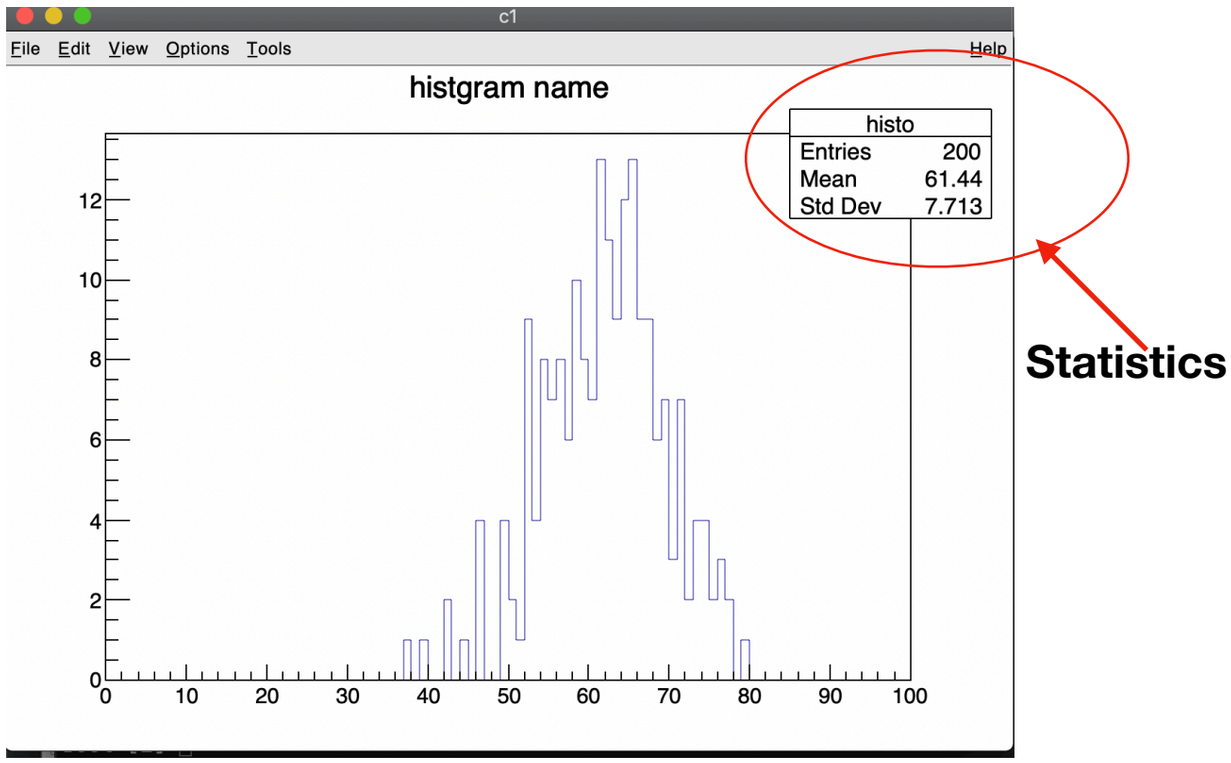
To study the performance of students, let’s plot a Histogram using root

Histogram

```
//To plot the histogram for the marks secured by the students
#include <iostream>
#include <fstream>
#include "TH1I.h"

void MyFistHisto(){

  ifstream inputFile("marks.dat"); // to open the data file

  //Define the histogram with object named histo
  TH1I *histo = new TH1I("histo", "histgram name", 100, 0, 100);

  string sFistLine;
  getline( inputFile, sFistLine);

  int slN0 =0, marks = 0;

  while( inputFile >> slN0 >> marks ){
    //for(int i = 0; i < 200; i++){

    histo->Fill( marks );
  }// while //for

  //plot the histogram--
  histo->Draw();

  inputFile.close();

}
```

```
Nirbhays-MBP:~ nirbhay$ root -l MyFistHisto.C
root [0]
Processing MyFistHisto.C...
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas
with name c1
root [1] .q
```

Output



Statistics

Where the axis title? Histogram name? etc.

Can be done by using the member functions of TH1 class.
Next slide

Histogram

Go to TH1 class for more details (https://root.cern.ch/doc/master/classTH1I.html)

Public Member Functions

	TH1I () Constructor. More...
	TH1I (const char ***name**, const char *title, **Int_t** nbinsx, **Double_t** xlow, **Double_t** xup) Create a 1-Dim histogram with fix bins of type integer (see **TH1::TH1** for explanation of parameters) More...
	TH1I (const char ***name**, const char *title, **Int_t** nbinsx, const **Float_t** *xbins) Create a 1-Dim histogram with variable bins of type integer (see **TH1::TH1** for explanation of parameters) More...
	TH1I (const char ***name**, const char *title, **Int_t** nbinsx, const **Double_t** *xbins) Create a 1-Dim histogram with variable bins of type integer (see **TH1::TH1** for explanation of parameters) More...
	TH1I (const **TH1I** &h1i) Copy constructor. More...
virtual	**~TH1I** () Destructor. More...

ROOT 6.21/01
Reference Guide

Search

TGraph2D
TGraph2DErrors
TGraphAsymmErrors
TGraphBentErrors
TGraphDelaunay
TGraphDelaunay2D
TGraphErrors
TGraphMultiErrors
TGraphSmooth
TGraphTime
TH1C
TH1D
TH1F
TH1I
TH1S
TH2C
TH2D
TH2F
TH2I
TH2Poly
TH2PolyBin
TH2S
TH3
TH3C
TH3D
TH3F

TH1I Member List

This is the complete list of members for TH1I, including all inherited members.

AbstractMethod(const char *method) const	**TObject**	
Add(TF1 *h1, Double_t c1=1, Option_t *option="")	**TH1**	virtual
Add(const TH1 *h1, Double_t c1=1)	**TH1**	virtual
Add(const TH1 *h, const TH1 *h2, Double_t c1=1, Double_t c2=1)	**TH1**	virtual
AddAt(Int_t c, Int_t i)	**TArrayI**	
AddBinContent(Int_t bin)	**TH1I**	virtual
AddBinContent(Int_t bin, Double_t w)	**TH1I**	virtual
AddDirectory(Bool_t add=kTRUE)	**TH1**	static
AddDirectoryStatus()	**TH1**	static
Adopt(Int_t n, Int_t *array)	**TArrayI**	
AndersonDarlingTest(const TH1 *h2, Option_t *option="") const	**TH1**	virtual
AndersonDarlingTest(const TH1 *h2, Double_t &advalue) const	**TH1**	virtual
AppendPad(Option_t *option="")	**TObject**	virtual
At(Int_t i) const	**TArrayI**	inline
AutoP2FindLimits(Double_t min, Double_t max)	**TH1**	protected virtual
AutoP2GetBins(Int_t n)	**TH1**	inline protected static
AutoP2GetPower2(Double_t x, Bool_t next=kTRUE)	**TH1**	inline protected static
BoundsOk(const char *where, Int_t at) const	**TArray**	inline protected
Browse(TBrowser *b)	**TH1**	virtual
BufferEmpty(Int_t action=0)	**TH1**	virtual
BufferFill(Double_t x, Double_t w)	**TH1**	protected virtual
CanExtendAllAxes() const	**TH1**	virtual
CheckAxisLimits(const TAxis *a1, const TAxis *a2)	**TH1**	protected static

SetLineStyle(Style_t lstyle)	**TAttLine**	inline virtual
SetLineWidth(Width_t lwidth)	**TAttLine**	inline virtual
SetMarkerAttributes()	**TAttMarker**	virtual
SetMarkerColor(Color_t mcolor=1)	**TAttMarker**	inline virtual
SetMarkerColorAlpha(Color_t mcolor, Float_t malpha)	**TAttMarker**	virtual
SetMarkerSize(Size_t msize=1)	**TAttMarker**	inline virtual
SetMarkerStyle(Style_t mstyle=1)	**TAttMarker**	inline virtual
SetMaximum(Double_t maximum=-1111)	**TH1**	inline virtual
SetMinimum(Double_t minimum=-1111)	**TH1**	inline virtual
SetName(const char *name)	**TH1**	virtual
SetNameTitle(const char *name, const char *title)	**TH1**	virtual
SetNdivisions(Int_t n=510, Option_t *axis="X")	**TH1**	virtual
SetNormFactor(Double_t factor=1)	**TH1**	inline virtual
SetObjectStat(Bool_t stat)	**TObject**	static
SetOption(Option_t *option=" ")	**TH1**	inline virtual
SetStatOverflows(EStatOverflows statOverflows)	**TH1**	inline
SetStats(Bool_t stats=kTRUE)	**TH1**	virtual
SetTickLength(Float_t length=0.02, Option_t *axis="X")	**TH1**	virtual
SetTitle(const char *title)	**TH1**	virtual
SetTitleFont(Style_t font=62, Option_t *axis="X")	**TH1**	virtual
SetTitleOffset(Float_t offset=1, Option_t *axis="X")	**TH1**	virtual
SetTitleSize(Float_t size=0.02, Option_t *axis="X")	**TH1**	virtual
SetUniqueID(UInt_t uid)	**TObject**	virtual
SetXTitle(const char *title)	**TH1**	inline virtual
SetYTitle(const char *title)	**TH1**	inline virtual
SetZTitle(const char *title)	**TH1**	inline virtual
ShowBackground(Int_t niter=20, Option_t *option="same")	**TH1**	virtual
ShowPeaks(Double_t sigma=2, Option_t *option="", Double_t threshold=0.05)	**TH1**	virtual

Setter function to set values, Getter function to get values

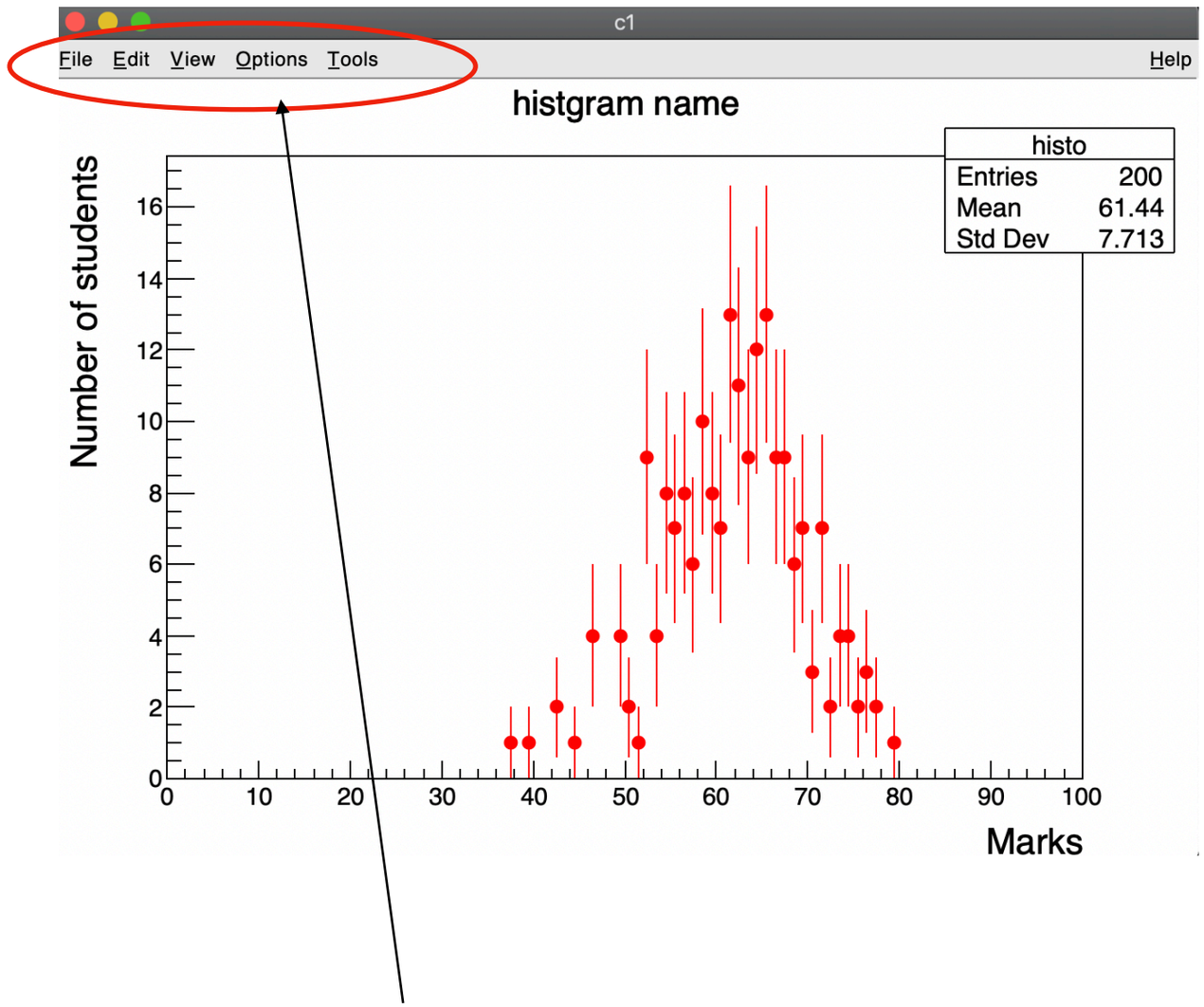
Histogram

```
/#include <iostream>
#include <fstream>
#include "TH1I.h"

void MyFistHisto(){

  ifstream inputFile("marks.dat"); // to open the data file

  //Define the histogram with object named histo
  TH1I *histo = new TH1I("histo", "histgram name", 100, 0, 100);
  histo->Sumw2();// for statistical error
  histo->SetXTitle("Marks"); //To set the x axis title
  histo->SetYTitle("Number of students"); //To set the y axis title
  histo->SetTitleSize(0.05,"xy"); //to set title size
  histo->SetMarkerStyle(20); //to set marker style circle, box, etc.
  histo->SetMarkerColor(2); //to set marker color: red, blue, black, etc
  histo->SetMarkerSize(1.); //set marker size
  histo->SetLineColor(2); //set line color

  string sFistLine;
  getline( inputFile, sFistLine);

  int slN0, marks;
  while( inputFile >> slN0 >> marks ){
    //for(int i = 0; i < 200; i++){

    histo->Fill( marks );
  }// while //for

  //plot the histogram--
  histo->Draw();

  inputFile.close();

}
```

Note the use of different member function of TH1

```
Nirbhays-MBP:~ nirbhay$ root -l MyFistHisto.C
root [0]
Processing MyFistHisto.C...
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas
with name c1
root [1] .q
```

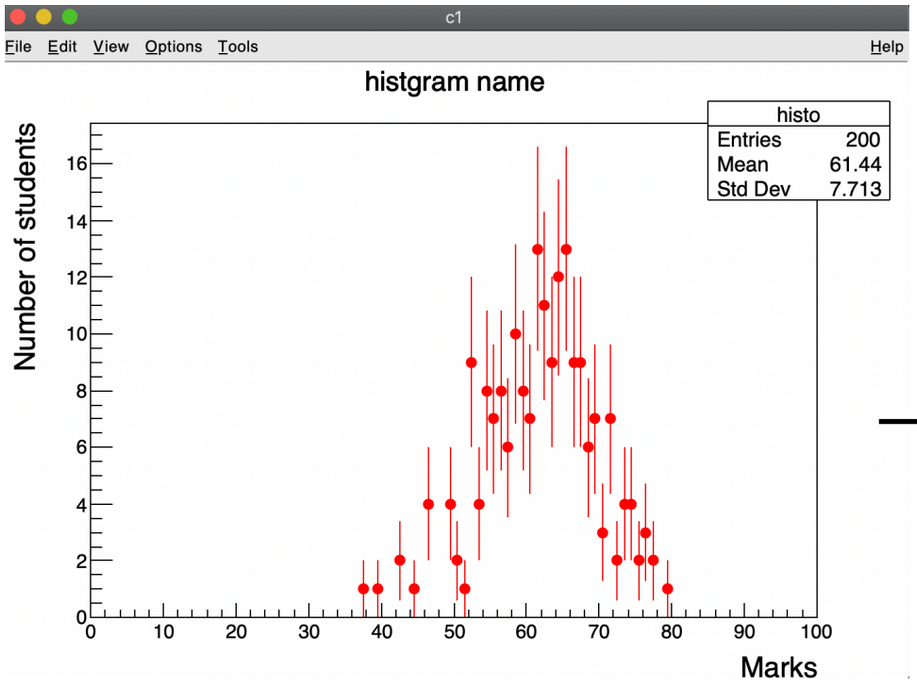
Output

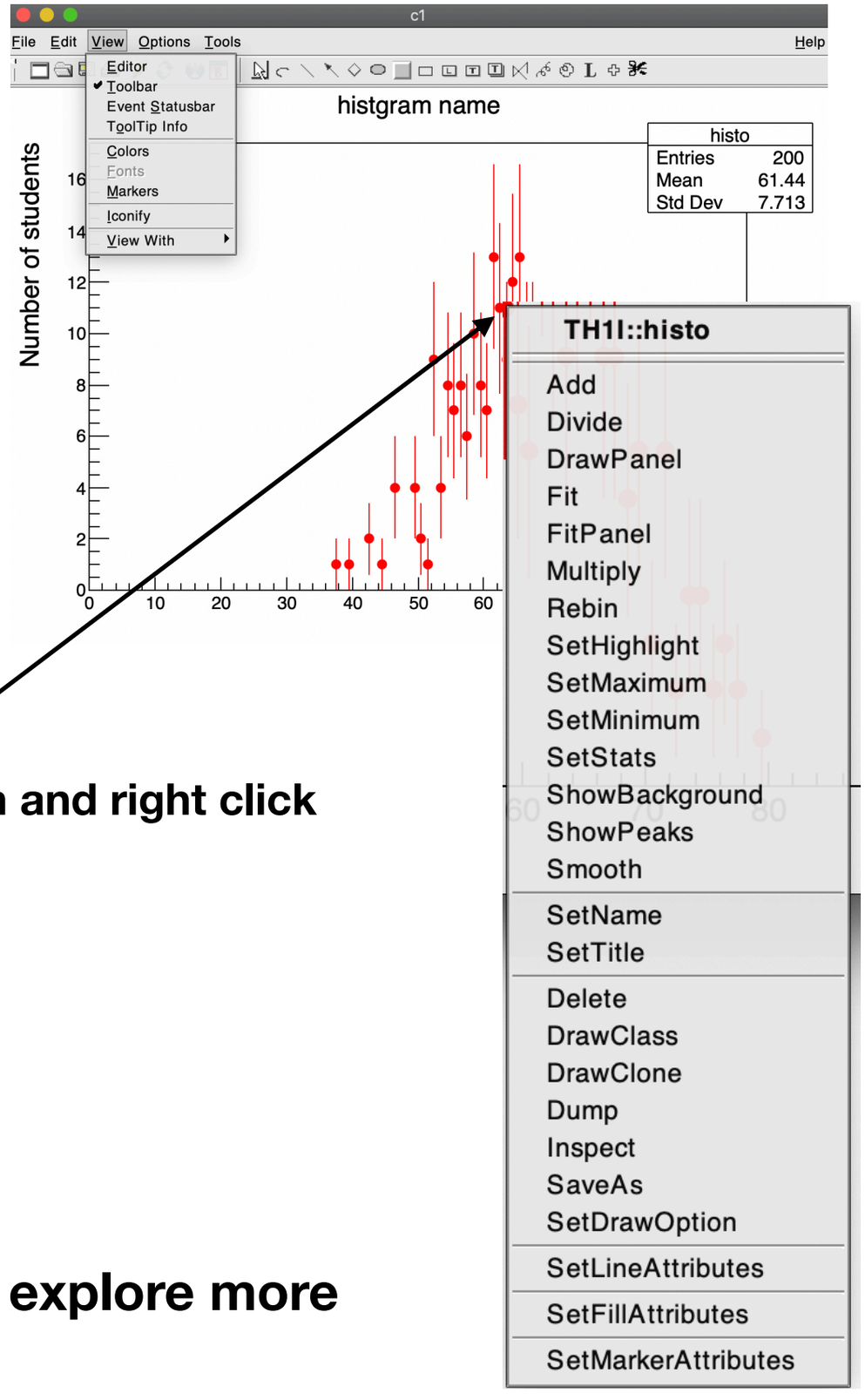


Further editing can be done from the toolbar

Histogram

Editing options using GUI





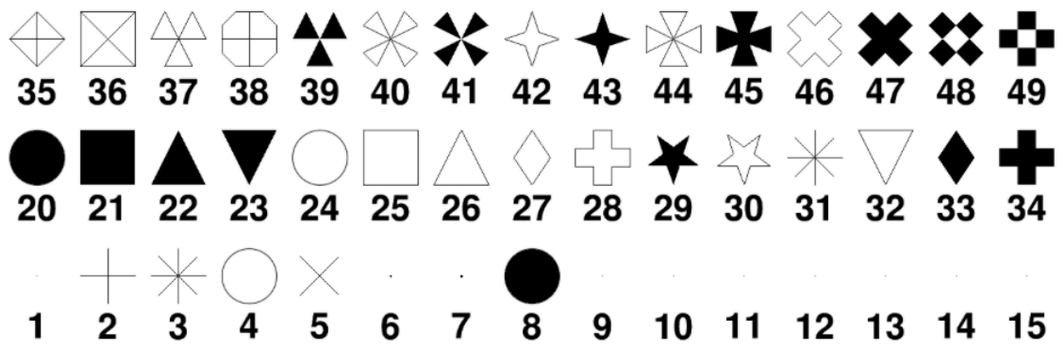
Put the cursor on the histogram and right click

Play with it and explore more

Histogram

Some color options: class TAttMarker

Different marker types



- **Similar exercise can be done for 2-D, 3-D and n-dimensional histogram.**
- **N-dim histogram is supported by THnSparse class (useful for multi-dimensional analysis of data.**
- **Two histograms can be added, subtracted, multiplied and divided by using ROOT.**
- **Highly efficient in big data analysis.**
- **Many visual options are available for 2-D and 3-D histos (like contur, lego, surface, color coded**
- **For more learning: https://root.cern/doc/master/group__tutorial__hist.html (also inside the directory: ROOT/tutorials/hist**

Graph

Graph plotting is supported by TGraph class

Public Member Functions

TGraph ()
Graph default constructor. More...

TGraph (**Int_t n**)
Constructor with only the number of points set the arrays x and y will be set later. More...

TGraph (**Int_t n**, const **Int_t** ***x**, const **Int_t** ***y**)
Graph normal constructor with ints. More...

TGraph (**Int_t n**, const **Float_t** ***x**, const **Float_t** ***y**)
Graph normal constructor with floats. More...

← **Use this for our example code**

TGraph (**Int_t n**, const **Double_t** ***x**, const **Double_t** ***y**)
Graph normal constructor with doubles. More...

TGraph (const **TGraph** &**gr**)
Copy constructor for this graph. More...

TGraph (const **TVectorF** &vx, const **TVectorF** &vy)
Graph constructor with two vectors of floats in input A graph is build with the X coordinates taken from vx and Y coord from vy The number of points in the graph is the minimum of number of points in vx and vy. More...

TGraph (const **TVectorD** &vx, const **TVectorD** &vy)
Graph constructor with two vectors of doubles in input A graph is build with the X coordinates taken from vx and Y coord from vy The number of points in the graph is the minimum of number of points in vx and vy. More...

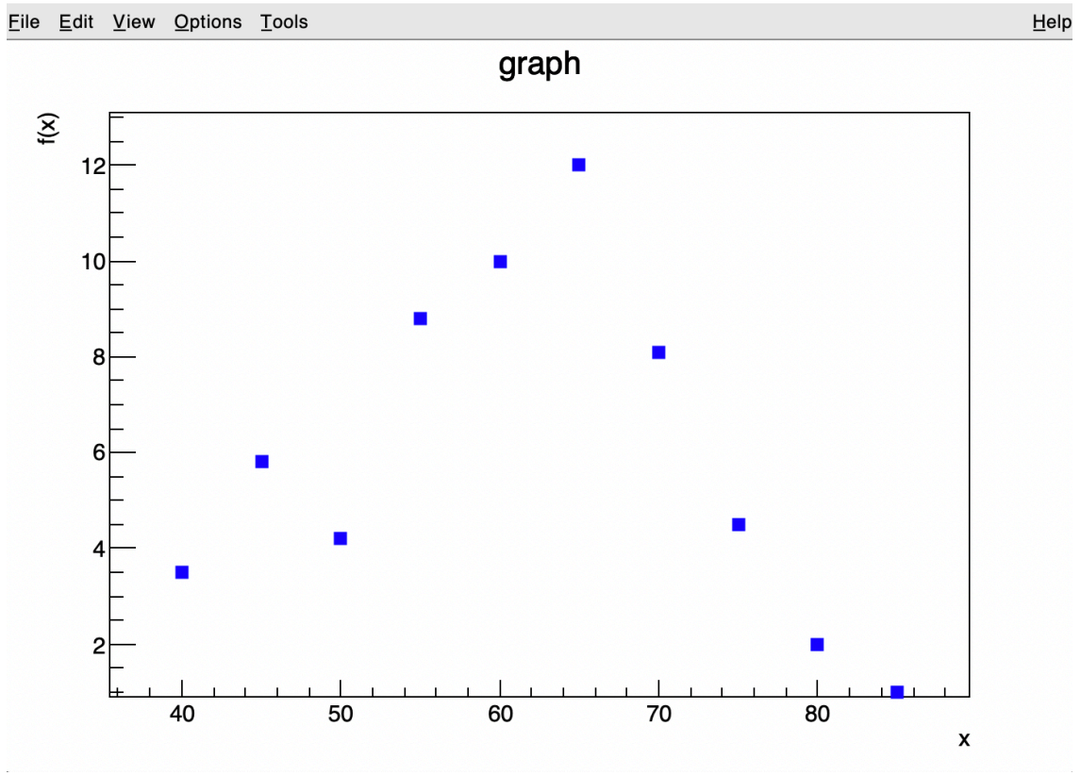
TGraph (const **TH1** ***h**)
Graph constructor importing its parameters from the **TH1** object passed as argument. More...

TGraph (const **TF1** ***f**, **Option_t** *option="")
Graph constructor importing its parameters from the **TF1** object passed as argument. More...

Many ways to define TGraph

Graph

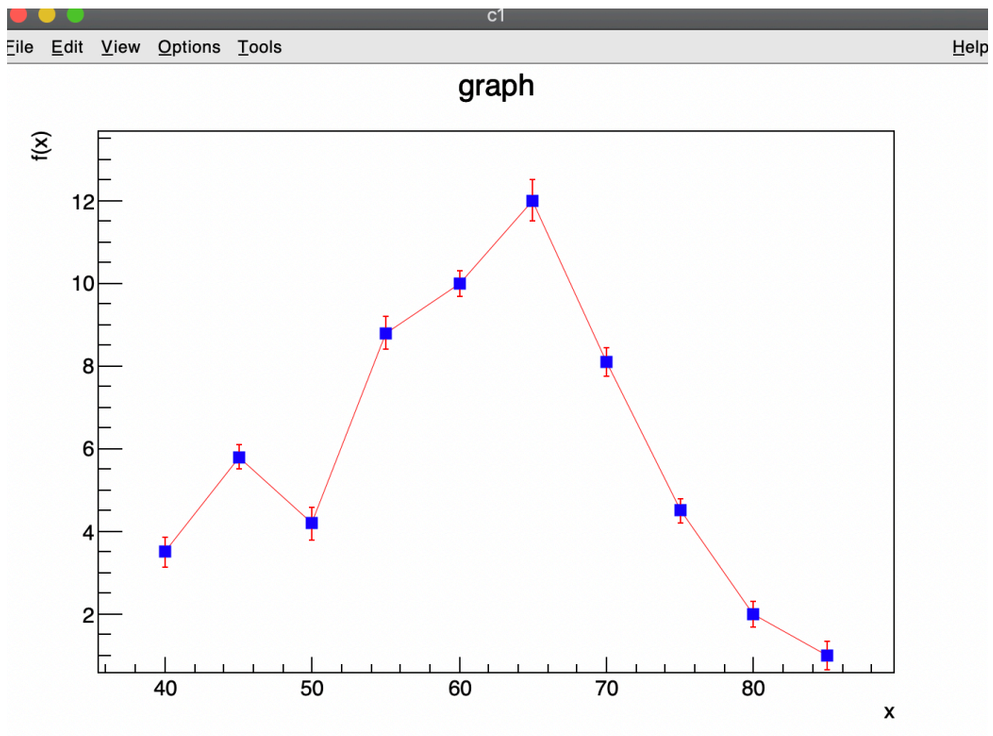
```
#include <iostream>
#include <fstream>
#include "TGraph.h"

void MyFirstGraph(){


  const int kDim = 10;
  float xValue[kDim] = { 40., 45., 50., 55., 60., 65., 70., 75., 80., 85. };
  float yValue[kDim] = {  3.5, 5.8, 4.2, 8.8, 10., 12., 8.1, 4.5, 2., 1. };

  TGraph *gr = new TGraph ( kDim, xValue, yValue);
  gr->SetTitle("graph; x; f(x)"); //To set the titles, x axis, y axis
  gr->SetMarkerStyle(21); //to set marker style circle, box, etc.
  gr->SetMarkerColor(4); //to set marker color: red, blue, black, etc
  gr->SetMarkerSize(1.); //set marker size
  gr->SetLineColor(2); //set line color
  gr->Draw("AP");

}
```



Graph

With error bars:

```
#include <iostream>
#include <fstream>
#include "TGraph.h"

void MyFirstGraph(){


  const int kDim = 10;
  float xValue[kDim] = { 40., 45., 50., 55., 60., 65., 70., 75., 80., 85. };
  float yValue[kDim] = {  3.5, 5.8, 4.2, 8.8, 10., 12., 8.1, 4.5, 2., 1. };
  float xValueError[kDim] = { 0., 0., 0., 0., 0., 0., 0., 0., 0., 0. };
  float yValueError[kDim] = {  0.35, 0.3, 0.4, 0.4, 0.3, 0.5, 0.35, 0.3, 0.3, 0.35 };

  //TGraph *gr = new TGraph ( kDim, xValue, yValue);
  TGraphErrors *gr = new TGraphErrors ( kDim, xValue, yValue, xValueError, yValueError);
  gr->SetTitle("graph; x; f(x)"); //To set the titles, x axis, y axis
  gr->SetMarkerStyle(21); //to set marker style circle, box, etc.
  gr->SetMarkerColor(4); //to set marker color: red, blue, black, etc
  gr->SetMarkerSize(1.); //set marker size
  gr->SetLineColor(2); //set line color
  gr->Draw("ALP");

}
```



Most of the editing functions are similar as histogram

Input/Output, file and data handling