

(C++) ROOT & ALICE Data Analysis (++) More ?!!)

Indranil Das

indranil.das@cern.ch

Outline : Life cycle of EHEP PhD student

- 1 C++ language
- 2 ROOT : HEP analysis tool
- 3 AliRoot : ALICE Analysis Software
- 4 Various

Handson tests

- 1 ROOT
- 2 Checkpoints
- 3 ALICE Analysis tests

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect "all" output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply .!pwd and pwd()
- 8 Try tab completion with edit("rootlogon.C")
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in "for" loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and +/++

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect "all" output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply .!pwd and pwd()
- 8 Try tab completion with edit("rootlogon.C")
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in "for" loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and +/+

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply .!pwd and pwd()
- 8 Try tab completion with edit("rootlogon.C")
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and +/++

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply .!pwd and pwd()
- 8 Try tab completion with edit("rootlogon.C")
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and +/++

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply .!pwd and pwd()
- 8 Try tab completion with edit(“rootlogon.C”)
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its’ detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and +/+

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply .!pwd and pwd()
- 8 Try tab completion with edit("rootlogon.C")
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and +/+

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : `first.C` vs `rootlogon.C`
- 15 Loading, unloading, running, compiling and `compile+run`
- 16 Compile in debug or optimized mode and `+/+`

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and `+ / ++`

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and `+ / ++`

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and `+ / ++`

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect "all" output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in "for" loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and `+ / ++`

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and `+ / ++`

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and compile+run
- 16 Compile in debug or optimized mode and `+ / ++`

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and `compile+run`
- 16 Compile in debug or optimized mode and `+ / ++`

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and `compile+run`
- 16 Compile in debug or optimized mode and `+ / ++`

Checkpoint I

- 1 Start root session with splash screen
- 2 Add, subtract, multiply, divide
- 3 Redirect “all” output of ROOT session to temp.out file
- 4 Print out global environments to output.txt
- 5 gROOT, gSystem, gRandom, gPad, gStyle
- 6 List the methods of a class
- 7 Go to \$ROOTSYS/tutorials, then apply `!.pwd` and `pwd()`
- 8 Try tab completion with `edit("rootlogon.C")`
- 9 Change the EDITOR environment and try again
- 10 Change back to earlier directory from \$ROOTSYS/tutorials
- 11 Create class TPoint and print its' detail information
- 12 Set and print the variables in “for” loop inside ROOT session
- 13 Dump the object member values
- 14 Unnamed and named script : first.C vs rootlogon.C
- 15 Loading, unloading, running, compiling and `compile+run`
- 16 Compile in debug or optimized mode and `+ / ++`

Checkpoint I

- 17 Write a class in a macro and run that prints "hello World"
- 18 How do you know the working directory inside macro ?
- 19 Where can you find earlier commands that have been applied during ROOT session ?
- 20 What is virtual function ? How to implement that in class ?
- 21 How to create abstract base class ?
- 22 Write code to create memory on stack and on heap
- 23 Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
- Spot the memory leak in the code that you have written before.
- What is code profiling ? How does it help to improve your code ?

Checkpoint I

- 17 Write a class in a macro and run that prints “hello World”
- 18 How do you know the working directory inside macro ?
- 19 Where can you find earlier commands that have been applied during ROOT session ?
- 20 What is virtual function ? How to implement that in class ?
- 21 How to create abstract base class ?
- 22 Write code to create memory on stack and on heap
- 23 Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
- Spot the memory leak in the code that you have written before.
- What is code profiling ? How does it help to improve your code ?

Checkpoint I

- 17 Write a class in a macro and run that prints “hello World”
- 18 How do you know the working directory inside macro ?
- 19 Where can you find earlier commands that have been applied during ROOT session ?
- 20 What is virtual function ? How to implement that in class ?
- 21 How to create abstract base class ?
- 22 Write code to create memory on stack and on heap
- 23 Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
- Spot the memory leak in the code that you have written before.
- What is code profiling ? How does it help to improve your code ?

Checkpoint I

- 17** Write a class in a macro and run that prints “hello World”
- 18** How do you know the working directory inside macro ?
- 19** Where can you find earlier commands that have been applied during ROOT session ?
- 20** What is virtual function ? How to implement that in class ?
- 21** How to create abstract base class ?
- 22** Write code to create memory on stack and on heap
- 23** Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
- Spot the memory leak in the code that you have written before.
- What is code profiling ? How does it help to improve your code ?

Checkpoint I

- 17 Write a class in a macro and run that prints "hello World"
- 18 How do you know the working directory inside macro ?
- 19 Where can you find earlier commands that have been applied during ROOT session ?
- 20 What is virtual function ? How to implement that in class ?
- 21 How to create abstract base class ?
- 22 Write code to create memory on stack and on heap
- 23 Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
- Spot the memory leak in the code that you have written before.
- What is code profiling ? How does it help to improve your code ?

Checkpoint I

- 17 Write a class in a macro and run that prints "hello World"
- 18 How do you know the working directory inside macro ?
- 19 Where can you find earlier commands that have been applied during ROOT session ?
- 20 What is virtual function ? How to implement that in class ?
- 21 How to create abstract base class ?
- 22 Write code to create memory on stack and on heap
- 23 Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
- Spot the memory leak in the code that you have written before.
- What is code profiling ? How does it help to improve your code ?

Checkpoint I

- 17** Write a class in a macro and run that prints “hello World”
 - 18** How do you know the working directory inside macro ?
 - 19** Where can you find earlier commands that have been applied during ROOT session ?
 - 20** What is virtual function ? How to implement that in class ?
 - 21** How to create abstract base class ?
 - 22** Write code to create memory on stack and on heap
 - 23** Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
 - Spot the memory leak in the code that you have written before.
 - What is code profiling ? How does it help to improve your code ?

Checkpoint I

- 17 Write a class in a macro and run that prints "hello World"
- 18 How do you know the working directory inside macro ?
- 19 Where can you find earlier commands that have been applied during ROOT session ?
- 20 What is virtual function ? How to implement that in class ?
- 21 How to create abstract base class ?
- 22 Write code to create memory on stack and on heap
- 23 Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
- Spot the memory leak in the code that you have written before.
- What is code profiling ? How does it help to improve your code ?

Checkpoint I

- 17 Write a class in a macro and run that prints "hello World"
- 18 How do you know the working directory inside macro ?
- 19 Where can you find earlier commands that have been applied during ROOT session ?
- 20 What is virtual function ? How to implement that in class ?
- 21 How to create abstract base class ?
- 22 Write code to create memory on stack and on heap
- 23 Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
- Spot the memory leak in the code that you have written before.
- What is code profiling ? How does it help to improve your code ?

Checkpoint I

- 17 Write a class in a macro and run that prints "hello World"
- 18 How do you know the working directory inside macro ?
- 19 Where can you find earlier commands that have been applied during ROOT session ?
- 20 What is virtual function ? How to implement that in class ?
- 21 How to create abstract base class ?
- 22 Write code to create memory on stack and on heap
- 23 Write an example memory leak code
- How to compile macro containing ROOT classes using g++ ?
- Spot the memory leak in the code that you have written before.
- What is code profiling ? How does it help to improve your code ?

Tree

- Arrange different types of objects and data types in single place.
- Formatted in such way such that accessing the entries is fast.
- While written to disk uses less disk resource

Checkpoint II

- 1 In case multiple files are opened, how the object can be written in the first file instead of last ?
 - 2 In which case the input ROOT file can be closed while you are still using the object stored into that file ?
 - 3 Write/read the event tree to/from ROOT file
 - 4 Now scan tree.root for fNtrack and fNvertex in ROOT session
 - 5 Next scan tree.root for fNTracks->fPx and fNvertex in ROOT session
 - 6 Draw fNTracks->fPx from tree.root in ROOT session using `tree->Draw("")`
 - 7 Draw fNTracks->fPx vs fNTracks->fPz from tree.root in ROOT session
 - 8 Draw fNTracks->fPx vs fNTracks->fPz for (fNvertex>5) in ROOT session
- Copy the tree.root of above example into tree1.root, tree2.root and tree3.root and read all three files using TChain in a macro.

Analysis Tutorial

The screenshot shows a web browser displaying the Indico event page for "Analysis Tutorial November 11". The event is scheduled for Friday, 11 Nov 2016, from 09:00 to 15:00 in Europe/Zurich at room 160-R-009 (CERN). The description states that the video recording of the tutorial can be found here. Under the "Videoconference Rooms" section, a room named "Analysis_Tutorial_November_11" is listed with a "Join" button. The event agenda includes two sessions: "Getting started with writing an analysis task" from 09:00 to 10:00, featuring speaker Redmer Alexander Bertens (NIKHEF National Institute for subatomic physics (NL)) and files "example.tar" and "bertens_analysis_L..."; and "aliBuild" from 10:00 to 10:30, described as a quick tutorial for getting started with ROOT and AliPhysics on your system, featuring speakers Dario Berzano (CERN) and Giulio Eulisse (CERN), with links to "Complete tutorial" and "Simplified tutorial".

■ <https://indico.cern.ch/event/586577/>

■ Or Open <http://alice-collaboration.web.cern.ch/> → Analysis → Tutorial

■ Download the example.tar

THANK YOU