

**KANDULA SRINIVASA REDDY MEMORIAL COLLEGE OF ENGINEERING
(AUTONOMOUS)**

KADAPA-516003. AP

(Approved by AICTE, Affiliated to JNTUA, Ananthapuramu, Accredited by NAAC)

(An ISO 9001-2008 Certified Institution)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING



VALUE ADDED COURSE

ON

“Object Oriented Programming through C++”

Resource Person : Smt. B. Manorama Devi, Dept. of CSE, KSRMCE

Course Coordinator: Smt. B. Manorama Devi, Dept. of CSE, KSRMCE

Duration: 17/01/2022 to 29/01/2022

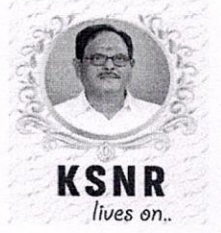


K.S.R.M. COLLEGE OF ENGINEERING (UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Lr./KSRMCE/AIML/2021-22/

Date: 08-01-2022

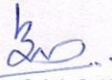
To
The Principal,
KSRMCE,
Kadapa.

Respected Sir,

Sub: Permission to Conduct Value added Course on “**Object Oriented Programming through C++**” from 17/01/2022 to 29/01/2022–Req- Reg.

The Department of Artificial Intelligence & Machine Learning is planning to offer a Value Added Course on “**Object Oriented Programming through C++**” to B. Tech. students. The course will be conducted from 17/01/2022 to 29/01/2022. In this regard, I kindly request you to grant permission to conduct Value Added Course.

Thanking you sir,


Yours faithfully

(Smt. B.Manorama Devi, Asst. Professor in CSE)

Permitted
V. S. S. Mm/5



K.S.R.M. COLLEGE OF ENGINEERING
(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Cr./KSRMCE/AIML/2021-22/

Date: 10/01/2022

Circular

The Department of Artificial Intelligence & Machine Learning is offering a Value Added Course on "Object Oriented Programming through C++" from **17/01/2022** to **29/01/2022** to B. Tech students. In this regard, interested students are requested to register their names for the Value Added Course with Course Coordinator.

<https://forms.gle/4tZeSFyqAQWQedD16>

For further information contact Course Coordinator.

Course Coordinator: Smt. B. Manorama Devi, Assistant Professor, Dept. of CSE.-KSRMCE.
Contact No: 9985725101

HOD

Dept. of AIML

Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA- 516 005. (A.P.)

Cc to:

IQAC-KSRMCE




/krmce.ac.in

Follow Us:



/krmceofficial

Registration for Value Added course on OOP's through C++

 sunil.j@ksrmce.ac.in (not shared) [Switch account](#)



* Required

Name *

Your answer

Roll Number *

Your answer

B.Tech Semester *

- I Sem
- II Sem
- III Sem
- IV Sem
- V Sem
- Other:



Registration for Value Added course on OOP's through C++

Branch *

- CSE
- ECE
- AIML
- EEE

Email *

Your answer

Submit

Clear form

Never submit passwords through Google Forms.

This form was created inside of KSRM College of Engineering. [Report Abuse](#)

Google Forms



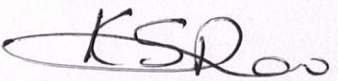
KSRM COLLEGE OF ENGINEERING, KADAPA

Registered Students List of Value Added Course on OOPs through C++

Timestamp	Name	Roll Number	B.Tech Semeste	Branch	Email
1-10-2022 14:38:17	Akuleti Pradeep Kumar	219Y1A3901		AI&ML	219Y1A3901@ksrmce.ac.in
1-10-2022 14:40:12	Annem Sharath Kumar Reddy	219Y1A3902		AI&ML	219Y1A3902@ksrmce.ac.in
1-10-2022 14:50:32	Attar Mohammed Sufiyan	219Y1A3903		AI&ML	219Y1A3903@ksrmce.ac.in
1-10-2022 14:57:17	AvulaAmith	219Y1A3904		AI&ML	219Y1A3904@ksrmce.ac.in
1-11-2022 19:57:17	Bhumireddy Venkata Ravi Kumar Reddy	219Y1A3905		AI&ML	219Y1A3905@ksrmce.ac.in
1-11-2022 20:00:20	Boranapu Sai Tejaswini (W)	219Y1A3906		AI&ML	219Y1A3906@ksrmce.ac.in
1-11-2022 20:57:12	Chandragiri Khamaleshwar	219Y1A3907		AI&ML	219Y1A3907@ksrmce.ac.in
1-12-2022 5:54:21	Chappidi Asha Deepthi Reddy (W)	219Y1A3908		AI&ML	219Y1A3908@ksrmce.ac.in
1-12-2022 15:57:11	Chenna Venkatesh	219Y1A3909		AI&ML	219Y1A3909@ksrmce.ac.in
1-12-2022 15:57:20	Dappella Vishnu Teja	219Y1A3910		AI&ML	219Y1A3910@ksrmce.ac.in
1-13-2022 8:21:10	Deepak Reddy Sirigireddy	219Y1A3911		AI&ML	219Y1A3911@ksrmce.ac.in
1-13-2022 19:32:43	Dodla Santhosh Reddy	219Y1A3912		AI&ML	219Y1A3912@ksrmce.ac.in
1-13-2022 19:21:23	DurgamRahamathunnisa (W)	219Y1A3913		AI&ML	219Y1A3913@ksrmce.ac.in
1-13-2022 20:57:20	GanganpalliNikhitha (W)	219Y1A3914		AI&ML	219Y1A3914@ksrmce.ac.in
1-14-2022 8:32:45	Gangireddy Sumanth Reddy	219Y1A3915		AI&ML	219Y1A3915@ksrmce.ac.in
1-14-2022 9:32:45	Gummalla Balaji Reddy	219Y1A3916		AI&ML	219Y1A3916@ksrmce.ac.in
1-14-2022 9:21:12	Kakumani Harika (W)	219Y1A3917		AI&ML	219Y1A3917@ksrmce.ac.in
1-14-2022 9:38:11	KanchamreddyJahnavi Reddy (W)	219Y1A3918		AI&ML	219Y1A3918@ksrmce.ac.in
1-14-2022 9:32:12	Kancharla Ganga Yatish	219Y1A3919		AI&ML	219Y1A3919@ksrmce.ac.in
1-14-2022 9:12:45	Kayapati Karthikeya Babruvahana	219Y1A3920		AI&ML	219Y1A3920@ksrmce.ac.in
1-14-2022 17:32:45	KrupakaranKarthikan	219Y1A3922		AI&ML	219Y1A3922@ksrmce.ac.in
1-14-2022 17:35:41	KuruvaMadhukrishna	219Y1A3923		AI&ML	219Y1A3923@ksrmce.ac.in
1-14-2022 17:37:47	Malireddy Sai Charan Reddy	219Y1A3924		AI&ML	219Y1A3924@ksrmce.ac.in
1-14-2022 17:39:21	Manchala Ranjith Kumar	219Y1A3925		AI&ML	219Y1A3925@ksrmce.ac.in
1-14-2022 17:45:45	Manega Harika(W)	219Y1A3926		AI&ML	219Y1A3926@ksrmce.ac.in
1-14-2022 17:45:45	Mayana Mohammed Ameer	219Y1A3927		AI&ML	219Y1A3927@ksrmce.ac.in
1-14-2022 17:45:45	Meruva Haritha (W)	219Y1A3928		AI&ML	219Y1A3928@ksrmce.ac.in
1-15-2022 21:25:05	Moghal Junaid Baig	219Y1A3929		AI&ML	219Y1A3929@ksrmce.ac.in
1-15-2022 22:46:05	Murthy Meghana (W)	219Y1A3930		AI&ML	219Y1A3930@ksrmce.ac.in
1-15-2022 23:21:05	Nagella Arjun	219Y1A3931		AI&ML	219Y1A3931@ksrmce.ac.in

1-15-2022 23:46:10	Nakka Guru Aakarsh	219Y1A3932		AI&ML	219Y1A3932@ksrmce.ac.in
1-15-2022 3:16:21	Nalipi Sridhar Reddy	219Y1A3933		AI&ML	219Y1A3933@ksrmce.ac.in
1-15-2022 22:46:05	Nallamalla Meghana (W)	219Y1A3934		AI&ML	219Y1A3934@ksrmce.ac.in
1-15-2022 22:46:05	NayanagariVasavi (W)	219Y1A3935		AI&ML	219Y1A3935@ksrmce.ac.in
1-15-2022 22:46:05	Obulareddy Gari Manasa (W)	219Y1A3936		AI&ML	219Y1A3936@ksrmce.ac.in
1-15-2022 22:46:05	P Likitha (W)	219Y1A3937		AI&ML	219Y1A3937@ksrmce.ac.in
1-15-2022 0:32:12	P Sowjanya (W)	219Y1A3938		AI&ML	219Y1A3938@ksrmce.ac.in
1-15-2022 22:46:05	Palleti Ram Sai Pravallika (W)	219Y1A3939		AI&ML	219Y1A3939@ksrmce.ac.in
1-15-2022 22:46:05	Pasala Naga Priyanka (W)	219Y1A3940		AI&ML	219Y1A3940@ksrmce.ac.in
1-15-2022 22:46:05	Patil Pradeep	219Y1A3942		AI&ML	219Y1A3942@ksrmce.ac.in
1-15-2022 22:46:05	Shaik Aseefulla	219Y1A3945		AI&ML	219Y1A3945@ksrmce.ac.in
1-15-2022 22:46:05	Shaik Asma Rehman (W)	219Y1A3946		AI&ML	219Y1A3946@ksrmce.ac.in
1-15-2022 22:46:05	Shaik Fysal Ahmed	219Y1A3947		AI&ML	219Y1A3947@ksrmce.ac.in
1-15-2022 22:46:05	Shaik Kattubadilmshad (W)	219Y1A3948		AI&ML	219Y1A394@ksrmce.ac.in
1-15-2022 22:46:05	Shaik Mohammed Gaffar Ali	219Y1A3951		AI&ML	219Y1A3951@ksrmce.ac.in
1-16-2022 17:45:45	Shaik Mohammed Saad	219Y1A3952		AI&ML	219Y1A3952@ksrmce.ac.in
1-16-2022 18:20:21	Shaik Shuaib	219Y1A3953		AI&ML	219Y1A3953@ksrmce.ac.in
1-16-2022 18:45:45	SirigireddyReddaiah	219Y1A3954		AI&ML	219Y1A3954@ksrmce.ac.in
1-16-2022 19:45:21	Somisetty Hemanth Kumar	219Y1A3955		AI&ML	219Y1A3955@ksrmce.ac.in
1-16-2022 19:25:13	Surya Sreenath	219Y1A3956		AI&ML	219Y1A3956@ksrmce.ac.in
1-16-2022 20:45:12	Syed Yezdan Ahamed	219Y1A3957		AI&ML	219Y1A3957@ksrmce.ac.in
1-16-2022 22:48:05	V Kuladeep	219Y1A3960		AI&ML	219Y1A3960@ksrmce.ac.in
1-16-2022 23:25:13	Vallepu Ajay	219Y1A3961		AI&ML	219Y1A3961@ksrmce.ac.in
1-16-2022 0:16:16	ValluruRuchitha (W)	219Y1A3962		AI&ML	219Y1A3962@ksrmce.ac.in
1-16-2022 2:45:45	Vankadhara Guru Jahnvi (W)	219Y1A3963		AI&ML	219Y1A3963@ksrmce.ac.in
1-16-2022 1:45:27	YadatiLakshmisravani (W)	219Y1A3964		AI&ML	219Y1A3964@ksrmce.ac.in
1-16-2022 21:45:45	YerugudipaduChennaKesava	219Y1A3965		AI&ML	219Y1A3965@ksrmce.ac.in


CO-ORDINATOR


HOD
Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
K. S. R. M. 516 001

SYLLABUS

OBJECTORIENTEDPROGRAMMING THROUGH C++

Course Description: This course introduce programming concept in C++. Students will be equipped with fundamental programming, Arrays, Functions, Exception etc.

Course Objectives:

1. Introduce the student to the concepts of C++ in computer science.
2. Acquire knowledge to make functions , Files atc

Course Outcomes:

1. Knowledge of programming language.
2. Be aware about OOP's concept..
3. Basic understanding on programming.

Course Content:

Unit-I

Introduction:

What is object oriented programming? Why do we need object-oriented. Programming characteristics of object-oriented languages. C and C++.

C++ Programming basics:

Output using cout. Directives. Input with cin. Type bool.The set wmanipulator. Type conversions.

Unit- II:

Functions:

Returning values from functions. Reference arguments.Overloadedfunction.Inlinefunction.Defaultarguments.Returningb yreference.

Object and Classes:

Makingsenseofcoreobjectconcepts(Encapsulation,Abstraction,Polymorphism ,Classes,MessagesAssociation,Interfaces)Implementation of class in C++, C++ Objects as physical object, C++object as data types constructor. Object as function arguments. The default copy constructor, returning object from function. Structures and classes. Classes objects and memory static class data. Const and classes.

Arrays and string arrays fundamentals: Arrays as class Member Data:

Arrays of object, string, The standard C++String class

Unit- III:

Operator overloading:

Overloadingunaryoperations.Overloadingbinaryoperators,dataconversion,pit fallsofoperatoroverloadingandconversionkeywords.ExplicitandMutable.

Inheritance:

Conceptofinheritance.Derivedclassandbasedclass.Derivedclassconstructors, memberfunction,inheritanceintheEnglishdistanceclass,classhierarchies,inheri tanceandgraphicsshapes,public and private inheritance, aggregation : Classes

within classes, inheritance and program development.

Unit- IV:

Pointer:

Addresses and pointers. The address of operator and pointer and arrays. Pointer and function pointer and C-types string. Memory management: New and Delete, pointer to objects, debugging pointers.

Virtual Function:

Virtual Function, friend function, Static function, Assignment and copy initialization, this pointer, dynamic type information.

Streams and Files:

Stream classes, Stream Errors, Disk File I/O with streams, file pointers, error handling in file I/O with member function, overloading the extraction and insertion operators, memory as a stream object, command line arguments, and printer output.

Unit- V:

Templates and Exceptions:

Function templates, Class templates Exceptions

The Standard Template Library:

Introduction algorithms, sequence containers, iterators, specialized iterators, associative containers, strong user-defined object, function objects.

Text Books:

1. C++ How to Program by H M Deitel and P J Deitel, 1998, Prentice Hall
2. Object Oriented Programming in Turbo C++ by Robert Lafore, 1994, The WAITE Group Press.
3. Programming with C++ By D Ravichandran, 2003, T.M.H

Reference Books:

1. Object oriented Programming with C++ by E Balagurusamy, 2001, Tata McGraw-Hill
2. Computing Concepts with C++ Essentials by Horstmann, 2003, John Wiley,
3. The Complete Reference in C++ By Herbert Schildt, 2002, TMH

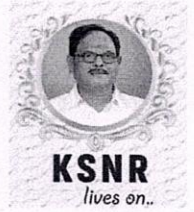


K.S.R.M. COLLEGE OF ENGINEERING (UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh, India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA,
Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Department of Artificial Intelligence & Machine Learning
Value added Course on Object Oriented Programming through C++
Schedule

S.No	Date	Time	Resource Person	Topic
1	17/01/2022	4PM to6PM	Smt. B. Manorama Devi	Inauguration Module I: Introduction C++ Programming basics
2	18/01/2022	4PM to6PM	Smt. B. Manorama Devi	Module II: Functions
3	19/01/2022	4PM to6PM	Smt. B. Manorama Devi	Functions
4	20/01/2022	4PM to6PM	Smt. B. Manorama Devi	Object and Classes
5	21/01/2022	4PM to6PM	Smt. B. Manorama Devi	Arrays and string arrays fundamentals
6	22/01/2022	4PM to6PM	Smt. B. Manorama Devi	Arrays and string arrays fundamentals
7	23/01/2022	2PM to5PM	Smt. B. Manorama Devi	Module III: Operator overloading
8	24/01/2022	4PM to6PM	Smt. B. Manorama Devi	Inheritance
9	25/01/2022	4PM to6PM	Smt. B. Manorama Devi	Inheritance
10	26/01/2022	4PM to6PM	Smt. B. Manorama Devi	Module IV: Pointer
11	27/01/2022	4PM to6PM	Smt. B. Manorama Devi	Virtual Function
12	28/01/2022	4PM to6PM	Smt. B. Manorama Devi	Streams and Files
13	29/01/2022	2PM to5PM	Smt. B. Manorama Devi	Module V: Template and Exceptions, Exam, Certificate Distribution and Vote of Thanks


Coordinator



HoD

Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA- 516 005. (A.P.)

49	219Y1A3955	Somisetty Hemanth Kumar	SK	SK	SK	SK	SK	SK	SK	SK	SK	SK	SK	SK	SK	SK	SK	SK	SK	SK
50	219Y1A3956	Surya Sreenath	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS
51	219Y1A3957	Syed Yezdan Ahamed	SY	SY	SY	SY	SY	SY	SY	SY	SY	SY	SY	SY	SY	SY	SY	SY	SY	SY
52	219Y1A3960	V Kuladeep	KU	KU	KU	KU	KU	KU	KU	KU	KU	KU	KU	KU	KU	KU	KU	KU	KU	KU
53	219Y1A3961	Vallepu Ajay	VA	VA	VA	VA	VA	VA	VA	VA	VA	VA	VA	VA	VA	VA	VA	VA	VA	VA
54	219Y1A3962	Valluru Ruchitha (W)	VR	VR	VR	VR	VR	VR	VR	VR	VR	VR	VR	VR	VR	VR	VR	VR	VR	VR
55	219Y1A3963	Vankadhara Guru Jahnvi (W)	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ	VJ
56	219Y1A3964	Yadati Lakshmisravani (W)	YL	YL	YL	YL	YL	YL	YL	YL	YL	YL	YL	YL	YL	YL	YL	YL	YL	YL
57	219Y1A3965	Yerugudipadu Chenna Kesava	YK	YK	YK	YK	YK	YK	YK	YK	YK	YK	YK	YK	YK	YK	YK	YK	YK	YK

7

30
Coordinator

KSRao

HOD

Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA- 516 0. (A.P.)



KSRM

COLLEGE OF ENGINEERING

(UGC - Autonomous)

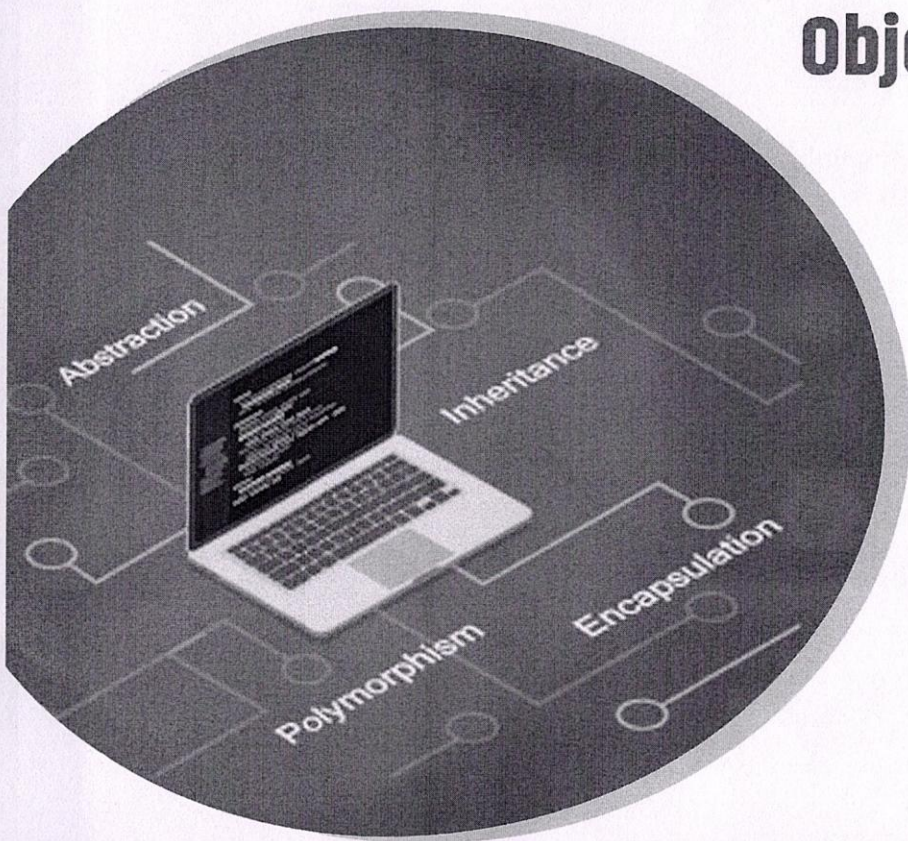
Kadapa, Andhra Pradesh, India- 516 005

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.



KSNR
lives on..

Value added course on Object Oriented Programming through C++



AI & ML



BDA Lab (PG-207)



17.01.2022 to 29.01.2022



4:00PM to 6:00PM

Resource Person

Smt. B. Manorama Devi

Asst. Professor in CSE

Coordinators

Smt. B. Manorama Devi

Asst. Professor in CSE

Dr. V.S.S. Murthy
(Principal)

Dr. Kandula Chandra Obul Reddy
(MD, KGI)

Smt. K.Rajeswari
(Correspondent, Secretary, Treasurer)

Sri K. Madan Mohan Reddy
(Vice - Chairman)

Sri K. Raja Mohan Reddy
(Chairman)

 [ksrcmceofficial](https://www.facebook.com/ksrcmceofficial)

 www.ksrcmce.ac.in

 8143731980, 8575697569



K.S.R.M. COLLEGE OF ENGINEERING

(AUTONOMOUS)

Pulivendala Road, Kadapa-516 005
Andhra Pradesh, India



Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

ACTIVITY REPORT

Value Added Course

On

Object Oriented Programming through C++
17/01/2022 to 29/01/2022 (4.00PM to 6.00PM)

Target Group	:	Students
Details of Participants	:	57 Students
Coordinator	:	Smt. B. Manorama Devi Asst. Prof, Dept. of CSE
Organizing Department	:	Artificial Intelligence & Machine Learning
Venue	:	BDA Lab (PG-207)

Description: Certification course on “Object Oriented Programming through C++” was organized by Dept. of CSE from 17/01/2022 to 29/01/2022. Smt. B. Manorama Devi acted as Course Coordinator and Resource person. The course is designed to provide complete knowledge of Object Oriented Programming through C++ and to enhance the programming skills of the students by giving practical assignments to be done in labs. The Star C++ Programming certification course provides an in-depth knowledge about C++.

C++ is a programming language with a special focus on the concepts of OOPs and their implementation. It has object-oriented features, which allow the programmer to create objects within the code. This makes programming easier, more efficient, and some would even say, more fun.

The course helps acquire a fundamental understanding of the OOPs concepts, input/output data management, arrays in C++, functions, classes, objects, pointers, and much more. The course has been designed with a uniform structured series of modules enumerating various pertinent concepts.

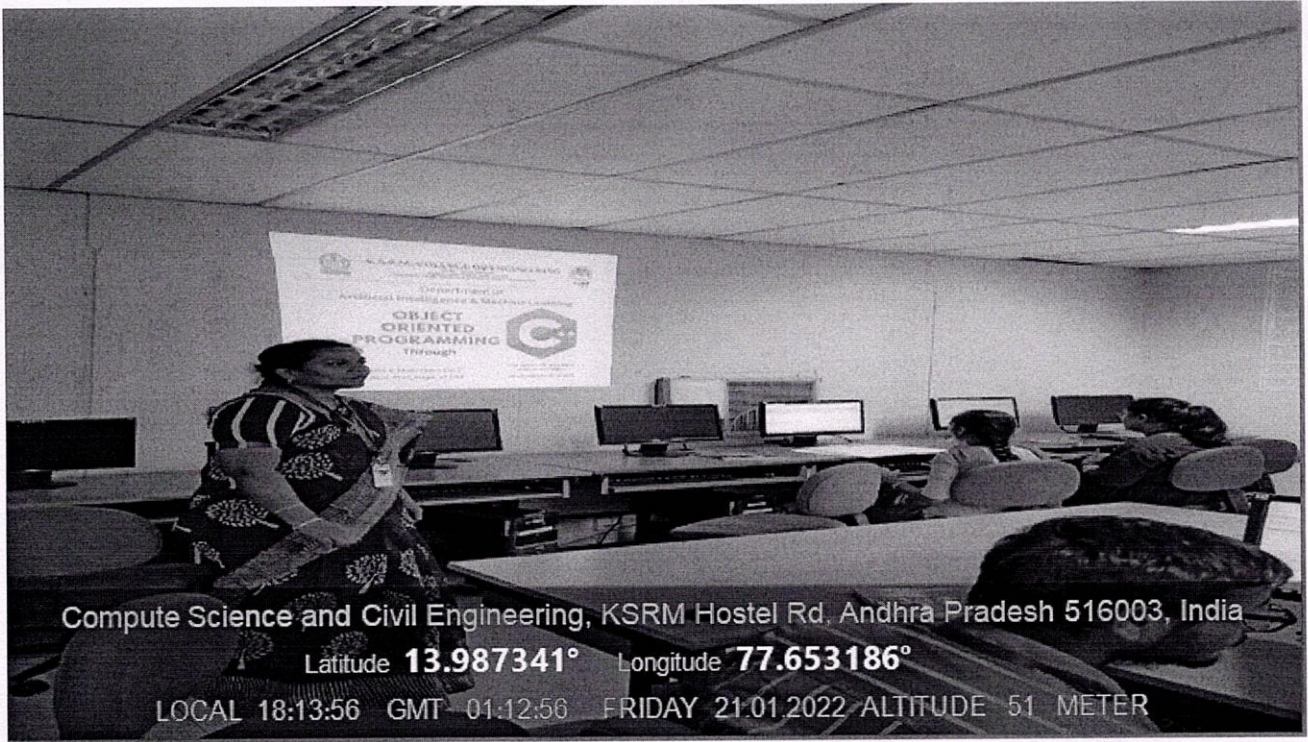
Photos:



Inauguration of the Program



Students attended for the course




Compute Science and Civil Engineering, KSRM Hostel Rd, Andhra Pradesh 516003, India

Latitude **13.987341°** Longitude **77.653186°**

LOCAL 18:13:56 GMT 01:12:56 FRIDAY 21-01-2022 ALTITUDE 51 METER

Lecture by Resource Person


Coordinator



HoD
Dr. K. SRINIVASA RAO, M.Tech., Ph.D.
Professor & HOD AIML
K.S.R.M. College of Engineering
(Autonomous)
KADAPA- 516 005. (A.P.)



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.



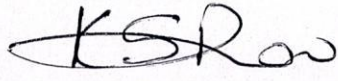
KSNR
lives on..

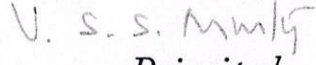
OBJECT ORIENTED PROGRAMMING THROUGH C++

CERTIFICATE OF PARTICIPATION

This is to certify that Mr/Mrs/Ms/Dr. 21941A3901 (A. PRADEEP KUMAR)
has participated in "Object Oriented Programming through C++" organized by Department
of AI & ML K.S.R.M College of Engineering , Kadapa, Andhrapradesh, during 17.01.2022 to 29.01.2022


Coordinator


HOD AIML


Principal



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.



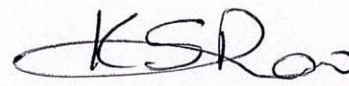
KSNR
lives on..

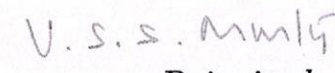
OBJECT ORIENTED PROGRAMMING THROUGH C++

CERTIFICATE OF PARTICIPATION

This is to certify that Mr/Mrs/Ms/Dr. G. SUMANTH REDDY (21941A3915)
has participated in "Object Oriented Programming through C++" organized by Department
of AI & ML K.S.R.M College of Engineering , Kadapa, Andhrapradesh, during 17.01.2022 to 29.01.2022


Coordinator


HOD AIML


Principal



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

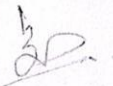


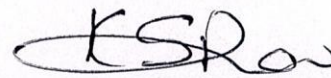
KSNR
lives on..

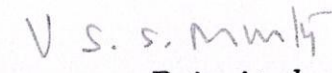
OBJECT ORIENTED PROGRAMMING THROUGH C++

CERTIFICATE OF PARTICIPATION


This is to certify that Mr/Mrs/Ms/Dr. G. PRAGATHI (219Y1A3944)
has participated in "Object Oriented Programming through C++" organized by Department
of AI & ML K.S.R.M College of Engineering , Kadapa, Andhrapradesh, during 17.01.2022 to 29.01.2022


Coordinator


HOD AIML


Principal

Feedback on Value Added course on OOP's through C++

 sunil.j@ksrmce.ac.in (not shared) [Switch account](#)



* Required

Name *

Your answer

Roll Number *

Your answer

Email *

Your answer

Branch *

Your answer



The objectives of the Value Added Course were met (Objective) *

- Excellent
- Good
- Satisfactory
- Poor

The content of the course was organized and easy to follow (Delivery) * *

- Excellent
- Good
- Satisfactory
- Poor

The Resource Persons were well prepared and able to answer any question *
(Interaction)

- Excellent
- Good
- Satisfactory
- Poor



The exercises/role play were helpful and relevant (Syllabus Coverage) * *

- Excellent
- Good
- Satisfactory
- Poor

The Value Added Course satisfy my expectation as a value added Programme *
(Course Satisfaction)

- Excellent
- Good
- Satisfactory
- Poor

Any Issues *

Your answer

Submit

Clear form

Never submit passwords through Google Forms.

This form was created inside of KSRM College of Engineering. [Report Abuse](#)

Google Forms



K.S.R.M. COLLEGE OF ENGINEERING

Department of Artificial Intelligence & Machine Learning, Feedback report on Value Added Course on OOP's through C++

Timestamp	Name	Roll Number	Email	Branch	The objectives of the Value Added Course were met	The content of the course was organized and easy to follow	The Resource Persons were well prepared and able to answer any	The exercises/role play were helpful and relevant (Syllabus)	The Value Added Course satisfy my expectation as a value	Any Issues
14:12:00	Akuleti Pradeep	219Y1A3901	srmce.ac.in	AIML	Excellent	Good	Satisfactory	Good	Good	No
14:13:33	Annem Sharath	219Y1A3902	219Y1A3902@k	AIML	Good	Satisfactory	Good	Excellent	Excellent	
14:16:23	Attar Mohammed	219Y1A3903	219Y1A3903@k	AIML	Excellent	Good	Good	Satisfactory	Good	
14:16:30	AvulaAmith	219Y1A3904	219Y1A3904@k	AIML	Good	Good	Excellent	Good	Good	
14:17:34	Bhumireddy Venkata	219Y1A3905	219Y1A3905@k	AIML	Satisfactory	Good	Excellent	Excellent	Good	
14:17:35	Boranapu Sai	219Y1A3906	219Y1A3906@k	AIML	Excellent	Good	Excellent	Good	Good	
14:17:40	Chandragiri	219Y1A3907	219Y1A3907@k	AIML	Excellent	Excellent	Good	Satisfactory	Good	
14:18:20	Chappidi Asha	219Y1A3908	219Y1A3908@k	AIML	Excellent	Satisfactory	Good	Excellent	Satisfactory	
14:18:22	Chenna Venkatesh	219Y1A3909	219Y1A3909@k	AIML	Excellent	Satisfactory	Excellent	Excellent	Satisfactory	
14:18:27	Dappella Vishnu Teja	219Y1A3910	219Y1A3910@k	AIML	Satisfactory	Good	Satisfactory	Good	Satisfactory	
15:19:40	Deepak Reddy	219Y1A3911	219Y1A3911@k	AIML	Excellent	Good	Satisfactory	Good	Good	
15:20:30	Dodla Santhosh	219Y1A3912	219Y1A3912@k	AIML	Good	Good	Good	Excellent	Good	
15:41:42	DurgamRahamathunn	219Y1A3913	219Y1A3913@k	AIML	Good	Good	Satisfactory	Excellent	Excellent	
16:20:22	GanganpalliNikhitha	219Y1A3914	219Y1A3914@k	AIML	Excellent	Good	Satisfactory	Good	Good	
16:21:44	Gangireddy Sumanth	219Y1A3915	219Y1A3915@k	AIML	Satisfactory	Satisfactory	Good	Good	Good	
16:32:23	Gummalla Balaji	219Y1A3916	219Y1A3916@k	AIML	Satisfactory	Excellent	Good	Excellent	Good	
16:32:46	Kakumani Harika (W)	219Y1A3917	219Y1A3917@k	AIML	Good	Good	Satisfactory	Satisfactory	Good	
16:40:20	KanchamreddyJahnavi	219Y1A3918	219Y1A3918@k	AIML	Good	Satisfactory	Good	Good	Good	
16:44:32	Kancharla Ganga	219Y1A3919	219Y1A3919@k	AIML	Good	Good	Excellent	Excellent	Excellent	
16:46:49	Kayapati Karthikeya	219Y1A3920	219Y1A3920@k	AIML	Good	Good	Satisfactory	Excellent	Excellent	
17:41:74	KrupakaranKarthikan	219Y1A3922	219Y1A3922@k	AIML	Good	Good	Satisfactory	Excellent	Excellent	NO
17:43:51	KuruvaMadhukrishna	219Y1A3923	219Y1A3923@k	AIML	Excellent	Good	Good	Good	Satisfactory	
17:41:52	Malireddy Sai Charan	219Y1A3924	219Y1A3924@k	AIML	Excellent	Good	Good	Excellent	Satisfactory	
17:43:35	Manchala Ranjith	219Y1A3925	219Y1A3925@k	AIML	Excellent	Good	Satisfactory	Excellent	Good	
17:51:54	Manega Harika(W)	219Y1A3926	219Y1A3926@k	AIML	Excellent	Satisfactory	Good		Good	NO
18:41:55	Mayana Mohammed	219Y1A3927	219Y1A3927@k	AIML	Good	Good	Good	Good	Good	

18:41:55	Meruva Haritha (W)	219Y1A3928	219Y1A3928@k	AIML	Excellent	Excellent	Good	Good	Good	
18:41:55	Murthy Meghana (W)	219Y1A3929	219Y1A3929@k	AIML	Good	Excellent		Excellent	Excellent	
18:41:59	Murthy Meghana (W)	219Y1A3930	219Y1A3930@k	AIML	Good	Satisfactory	Good	Good	Good	
18:44:59	Nagella Arjun	219Y1A3931	219Y1A3931@k	AIML	Good	Good	Good	Satisfactory	Good	
18:44:59	Nakka Guru Aakarsh	219Y1A3932	219Y1A3932@k	AIML	Good	Good	Satisfactory	Good	Satisfactory	
18:45:30	Nalipi Sridhar Reddy	219Y1A3933	219Y1A3933@k	AIML	Excellent	Excellent	Excellent	Satisfactory	Excellent	
18:45:42	Nallamalla Meghana	219Y1A3934	219Y1A3934@k	AIML	Good	Excellent	Excellent	Good	Good	
18:46:20	Nayanagari Vasavi (W)	219Y1A3935	219Y1A3935@k	AIML	Good	Good	Good	Satisfactory	Good	
18:46:21	Obulareddy Gari	219Y1A3936	219Y1A3936@k	AIML	Good	Good	Excellent	Good	Excellent	
18:46:21	P Likitha (W)	219Y1A3937	219Y1A3937@k	AIML	Good	Good	Good	Good	Excellent	NO
18:47:34	P Sowjanya (W)	219Y1A3938	219Y1A3938@k	AIML	Excellent	Good	Good	Satisfactory	Excellent	
18:47:37	Palleti Ram Sai	219Y1A3939	219Y1A3939@k	AIML	Good	Good	Excellent	Good	Good	
18:47:42	Pasala Naga Priyanka	219Y1A3940	219Y1A3940@k	AIML	Good	Good	Excellent	Satisfactory	Excellent	
18:47:50	Patil Pradeep	219Y1A3942	219Y1A3942@k	AIML	Excellent	Satisfactory	Excellent	Satisfactory	Excellent	
19:32:50	Shaik Aseefulla	219Y1A3945	219Y1A3945@k	AIML	Good	Satisfactory	Excellent	Satisfactory	Good	
19:40:21	Shaik Asma Rehman	219Y1A3946	219Y1A3946@k	AIML	Excellent	Excellent	Good	Good	Excellent	
19:50:50	Shaik Fysal Ahmed	219Y1A3947	219Y1A3947@k	AIML	Good	Good	Good	Excellent	Good	
19:54:21	Shaik	219Y1A3948	219Y1A394@ksr	AIML	Good	Good	Good	Good	Good	
19:60:74	Shaik Mohammed	219Y1A3951	219Y1A3951@k	AIML	Excellent	Good	Excellent	Excellent	Good	
19:61:75	Shaik Mohammed	219Y1A3952	219Y1A3952@k	AIML	Satisfactory	Excellent	Excellent	Satisfactory	Good	
19:61:75	Shaik Shuaib	219Y1A3953	219Y1A3953@k	AIML	Excellent	Excellent	Excellent	Excellent	Excellent	
19:61:75	Sirigireddy Reddaiah	219Y1A3954	219Y1A3954@k	AIML	Excellent	Good	Good	Good	Satisfactory	NO
20:01:21	Somisetty Hemanth	219Y1A3955	219Y1A3955@k	AIML	Excellent	Satisfactory	Good	Good	Good	
20:01:21	Surya Sreenath	219Y1A3956	219Y1A3956@k	AIML	Excellent	Good	Excellent	Good	Excellent	
20:12:19	Syed Yezdan Ahamed	219Y1A3957	219Y1A3957@k	AIML	Good	Good	Satisfactory	Good	Good	
20:30:13	V Kuladeep	219Y1A3960	219Y1A3960@k	AIML	Good	Good	Good	Good	Good	
21:10:45	Vallepu Ajay	219Y1A3961	219Y1A3961@k	AIML	Excellent	Good	Excellent	Good	Good	
21:24:21	Valluru Ruchitha (W)	219Y1A3962	219Y1A3962@k	AIML	Good	Satisfactory	Good	Excellent	Good	
21:30:44	Vankadhara Guru	219Y1A3963	219Y1A3963@k	AIML	Good	Satisfactory	Good	Good	Good	
21:34:87	Yadati Lakshmisravani	219Y1A3964	219Y1A3964@k	AIML	Good	Excellent	Good	Satisfactory	Good	
21:36:21	Yerugudipadu Chenna	219Y1A3965	219Y1A3965@k	AIML	Satisfactory	Excellent	Good	Good	Excellent	

COORDINATOR

Dr. K. SRINIVASA RAO M.Tech., Ph.D.
Professor & HOD AIML

K. J. S. M. College of Engineering

(Autonomous)

K. J. S. M. College of Engineering

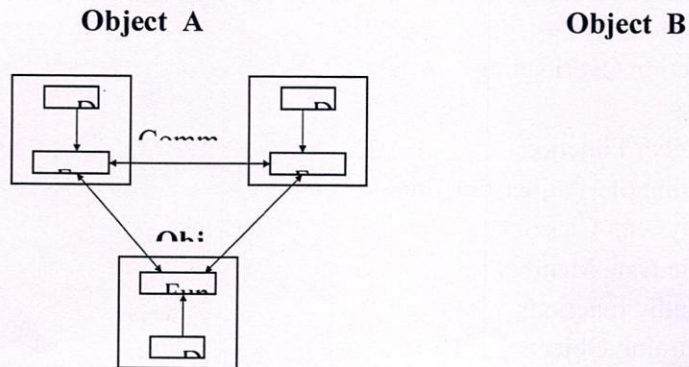
CONTENTS

Lecture 01:	Introduction
Lecture 02:	Object Oriented Programming
Lecture 03:	BASIC CONCEPTS OF OBJECTS ORIENTED PROGRAMMING
Lecture 04:	BENEFITS OF OOP
Lecture 05:	Basics of C++
Lecture 06:	Tokens
Lecture 07:	Basic Data types in C++
Lecture 08:	Symbolic Constant
Lecture 09:	Operators
Lecture 10:	Control Structures
Lecture 11:	Functions in C++
Lecture 12:	Function Overloading
Lecture 13:	Class
Lecture 14:	Member Function
Lecture 15:	Nesting of Member function
Lecture 16:	Array with Class
Lecture 17:	Static Data Member
Lecture 18:	Friendly functions
Lecture 19:	Returning Objects
Lecture 20:	Constructors
Lecture 21:	Destructors
Lecture 22 & 23:	Operator Overloading
Lecture 24:	Type Conversion
Lecture 25:	Class to Basic type
Lecture 26:	Inheritance
Lecture 27:	Multilevel Inheritance
Lecture 28:	Hierarchical Inheritance
Lecture 29:	Virtual Base Class
Lecture 30:	Polymorphism
Lecture 31:	Virtual functions
Lecture 32:	Pure Virtual Functions
Lecture 33:	C++ function overriding
Lecture 34:	Exception Handling
Lecture 35:	Array reference out of bound
Lecture 36:	Containership in C++
Lecture 37:	Template
Lecture 38:	Class Template
Lecture 39:	Virtual destructors
Lecture 40:	Managing Console I/O
Lecture 41:	Namespaces
Lecture 42:	New & Delete Operators

LECTURE-2

Object Oriented Programming

“Object oriented programming as an approach that provides a way of modularizing programs by creating partitioned memory area for both data and functions that can be used as templates for creating copies of such modules on demand”.



Features of the Object Oriented programming

1. Emphasis is on doing rather than procedure.
2. programs are divided into what are known as objects.
3. Data structures are designed such that they characterize the objects.
4. Functions that operate on the data of an object are tied together in the data structure.
5. Data is hidden and can't be accessed by external functions.
6. Objects may communicate with each other through functions.
7. New data and functions can be easily added.
8. Follows bottom-up approach in program design.

UNIT - I

LECTURE-1

Introduction:

Programmers write instructions in various programming languages to perform their computation tasks such as:

- (i) Machine level Language
- (ii) Assembly level Language
- (iii) High level Language

Machine level Language :

Machine code or machine language is a set of instructions executed directly by a computer's central processing unit (CPU). Each instruction performs a very specific task, such as a load, a jump, or an ALU operation on a unit of data in a CPU register or memory. Every program directly executed by a CPU is made up of a series of such instructions.

Assembly level Language :

An assembly language (or assembler language) is a low-level programming language for a computer, or other programmable device, in which there is a very strong (generally one-to-one) correspondence between the language and the architecture's machine code instructions. Assembly language is converted into executable machine code by a utility program referred to as an assembler; the conversion process is referred to as assembly, or assembling the code.

High level Language :

High-level language is any programming language that enables development of a program in much simpler programming context and is generally independent of the computer's hardware architecture. High-level language has a higher level of abstraction from the computer, and focuses more on the programming logic rather than the underlying hardware components such as memory addressing and register utilization.

The first high-level programming languages were designed in the 1950s. Now there are dozens of different languages, including Ada , Algol, BASIC, COBOL, C, C++, JAVA, FORTRAN, LISP, Pascal, and Prolog. Such languages are considered high-level because they are closer to human languages and farther from machine languages. In contrast, assembly languages are considered low-level because they are very close to machine languages.

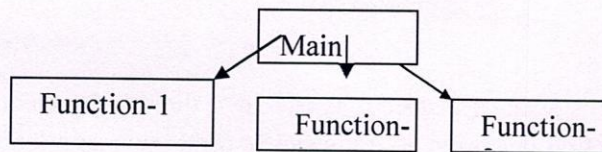
The high-level programming languages are broadly categorized in to two categories:

- (iv) Procedure oriented programming(POP) language.
- (v) Object oriented programming(OOP) language.

Procedure Oriented Programming Language

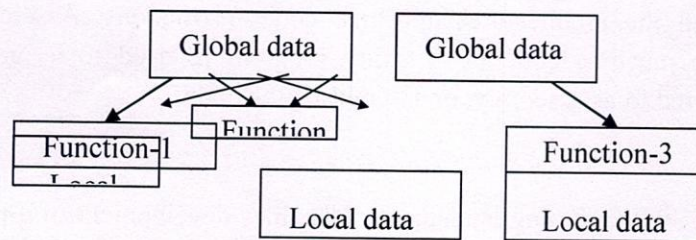
In the procedure oriented approach, the problem is viewed as sequence of things to be done such as reading , calculation and printing.

Procedure oriented programming basically consist of writing a list of instruction or actions for the computer to follow and organizing these instruction into groups known as functions.



The disadvantage of the procedure oriented programming languages is:

1. Global data access
2. It does not model real word problem very well
3. No data hiding



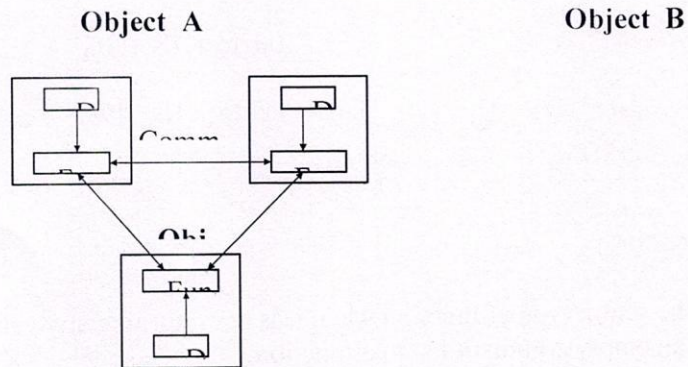
Characteristics of procedure oriented programming:

1. Emphasis is on doing things(algorithm)
2. Large programs are divided into smaller programs known as functions.
3. Most of the functions share global data
4. Data move openly around the system from function to function
5. Function transforms data from one form to another.
6. Employs top-down approach in program design

LECTURE-2

Object Oriented Programming

“Object oriented programming as an approach that provides a way of modularizing programs by creating partitioned memory area for both data and functions that can be used as templates for creating copies of such modules on demand”.



Features of the Object Oriented programming

1. Emphasis is on doing rather than procedure.
2. programs are divided into what are known as objects.
3. Data structures are designed such that they characterize the objects.
4. Functions that operate on the data of an object are tied together in the data structure.
5. Data is hidden and can't be accessed by external functions.
6. Objects may communicate with each other through functions.
7. New data and functions can be easily added.
8. Follows bottom-up approach in program design.

long int	4	-2147483648 to 2147483648
signed long int	4	-2147483648 to 2147483648
unsigned long int	4	0 to 4294967295
float	4	3.4E-38 to 3.4E+38
double	8	1.7E -308 to 1.7E +308
long double	10	3.4E-4932 to 1.1E+ 4932

The type void normally used for:

- 1) To specify the return type of function when it is not returning any value.
- 2) To indicate an empty argument list to a function.

Example:

```
Void function(void);
```

Another interesting use of void is in the declaration of generic pointer

Example:

```
Void *gp;
```

Assigning any pointer type to a void pointer without using a cast is allowed in both C and ANSI C. In ANSI C we can also assign a void pointer to a non-void pointer without using a cast to non void pointer type. This is not allowed in C ++.

Example:

```
void *ptr1;
```

```
void *ptr2;
```

Are valid statement in ANSI C but not in C++. We need to use a cast operator.

```
ptr2=(char * ) ptr1;
```

USER DEFINED DATA TYPES:

STRUCTERS AND CLASSES

We have used user defined data types such as struct, and union in C. While these more features have been added to make them suitable for object oriented programming. C++ also permits us to define

DATA ABSTRACTION :

Abstraction refers to the act of representing essential features without including the background details or explanations. Classes use the concept of abstraction and are defined as size, width and cost and functions to operate on the attributes.

DATA ENCAPSALATION :

The wrapping up of data and function into a single unit (called class) is known as encapsulation. The data is not accessible to the outside world and only those functions which are wrapped in the class can access it. These functions provide the interface between the objects data and the program.

INHERITENCE :

Inheritance is the process by which objects of one class acquire the properties of another class. In the concept of inheritance provides the idea of reusability. This means that we can add additional features to an existing class without modifying it. This is possible by designing a new class which will have the combined features of both the classes.

POLYMORPHISM:

Polymorphism means the ability to take more than one form. An operation may exhibit different instances. The behaviour depends upon the type of data used in the operation.

A language feature that allows a function or operator to be given more than one definition. The types of the arguments with which the function or operator is called determines which definition will be used.

Overloading may be operator overloading or function overloading.

It is able to express the operation of addition by a single operator say '+'. When this is possible you use the expression $x + y$ to denote the sum of x and y , for many different types of x and y ; integers, float and complex no. You can even define the + operation for two strings to mean the concatenation of the strings.

DYNAMIC BINDING :

Binding refers to the linking of a procedure call to the code to be executed in response to the call. Dynamic binding means the code associated with a given procedure call is not known until the time of the call at run-time. It is associated with a polymorphic reference depends upon the dynamic type of that reference.

1. Only alphabetic chars, digits and under score are permitted.
2. The name can't start with a digit.
3. Upper case and lower case letters are distinct.
4. A declared keyword can't be used as a variable name.

In ANSIC the maximum length of a variable is 32 chars but in c++ there is no bar.

LECTURE- 4

BENEFITS OF OOP:

Oop offers several benefits to both the program designer and the user. Object-oriented contributes to the solution of many problems associated with the development and quality of software products. The principal advantages are :

1. Through inheritance we can eliminate redundant code and extend the use of existing classes.
2. We can build programs from the standard working modules that communicate with one another, rather than having to start writing the code from scratch. This leads to saving of development time and higher productivity.
3. This principle of data hiding helps the programmer to build secure programs that can't be invaded by code in other parts of the program.
4. It is possible to have multiple instances of an object to co-exist with out any interference.
5. It is easy to partition the work in a project based on objects.
6. Object-oriented systems can be easily upgraded from small to large systems.
7. Message passing techniques for communication between objects makes the interface description with external systems much simpler.
8. Software complexity can be easily managed.

APPLICATION OF OOP:

The most popular application of oops up to now, has been in the area of user interface design such as windows. There are hundreds of windowing systems developed using oop techniques.

Real business systems are often much more complex and contain many more objects with complicated attributes and methods. Oop is useful in this type of applications because it can simplify a complex problem. The promising areas for application of oop includes.

1. Real – Time systems.
2. Simulation and modeling
3. Object oriented databases.
4. Hypertext,hypermedia and expertext.
5. AI and expert systems.
6. Neural networks and parallel programming.
7. Dicision support and office automation systems.
8. CIM / CAM / CAD system.

```
{
char name[30];
int age;
public:
    void getdata(void);
    void display(void);
};

void person :: getdata ( void )
{
    cout<<"enter name";
    cin>>name;
    cout<<"enter age";
    cin>>age;
}

void display()
{
    cout<<"\n name:"<<name;
    cout<<"\n age:"<<age;
}

int main()
{

person p;
p.getdata();
p.display();
return(0);
}
```

LECTURE-3

BASIC CONCEPTS OF OBJECTS ORIENTED PROGRAMMING

1. Objects
2. Classes
3. Data abstraction and encapsulation
4. Inheritance
5. Polymorphism
6. Dynamic binding
7. Message passing

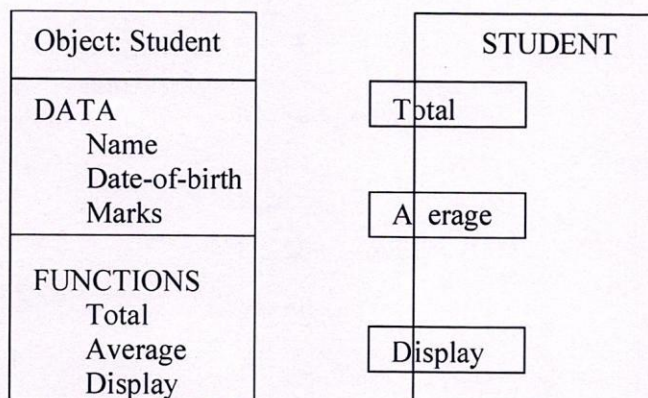
OBJECTS

Objects are the basic run-time entities in an object-oriented system. They may represent a person, a place, a bank account, a table of data or any item that the program must handle.

The fundamental idea behind object oriented approach is to combine both data and function into a single unit and these units are called objects.

The term objects means a combination of data and program that represent some real word entity. For example: consider an example named Amit; Amit is 25 years old and his salary is 2500. The Amit may be represented in a computer program as an object. The data part of the object would be (name: Amit, age: 25, salary: 2500)

The program part of the object may be collection of programs (retrive of data, change age, change of salary). In general even any user –defined type-such as employee may be used. In the Amit object the name, age and salary are called attributes of the object.



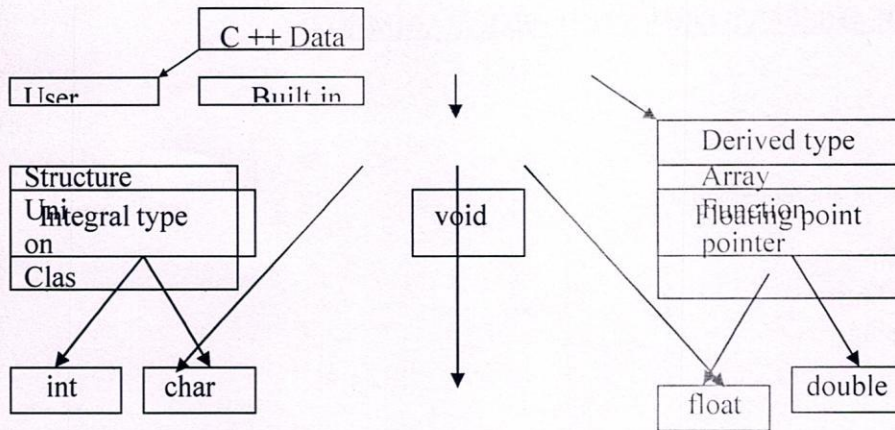
CLASS:

A group of objects that share common properties for data part and some program part are collectively called as class.

In C ++ a class is a new data type that contains member variables and member functions that operate on the variables.

Lecture-7

BASIC DATA TYPES IN C++



Both C and C++ compilers support all the built in types. With the exception of void the basic datatypes may have several modifiers preceding them to serve the needs of various situations. The modifiers signed, unsigned, long and short may be applied to character and integer basic data types. However the modifier long may also be applied to double.

Data types in C++ can be classified under various categories.

<u>TYPE</u>	<u>BYTES</u>	<u>RANGE</u>
char	1	-128 to 127
unsigned	1	0 to 255
signed char	1	-128 to 127
int	2	-32768 to 32768
unsigned int	2	0 to 65535
signed int	2	-32768 to 32768
short int	2	-32768 to 32768

LECTURE-6

TOKENS:

The smallest individual units in program are known as **tokens**. C++ has the following tokens.

- i. Keywords
- ii. Identifiers
- iii. Constants
- iv. Strings
- v. Operators

KEYWORDS:

The keywords implement specific C++ language feature. They are explicitly reserved identifiers and can't be used as names for the program variables or other user defined program elements. The keywords not found in ANSI C are shown in red letter.

C++ KEYWORDS:

Asm	double	new	Switch
Auto	else	operator	Template
Break	enum	private	This
Case	extern	protected	Throw
Catch	float	public	Try
Char	for	register	Typedef
Class	friend	return	Union
Const	goto	short	unsigned
Continue	if	signed	Virtual
Default	inline	sizeof	Void
Delete	long	struet	While

IDENTIFIERS:

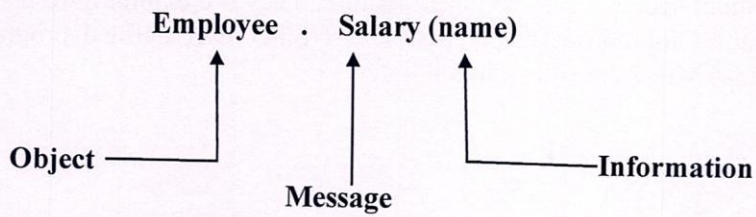
Identifiers refers to the name of variable , functions, array, class etc. created by programmer. Each language has its own rule for naming the identifiers.

The following rules are common for both C and C++.

MESSAGE PASSING :

An object oriented program consists of a set of objects that communicate with each other.

A message for an object is a request for execution of a procedure and therefore will invoke a function (procedure) in the receiving object that generates the desired result. Message passing involves specifying the name of the object, the name of the function (message) and information to be sent.



executed in any C++ program. For that same reason, it is essential that all C++ programs have a main function.

The word main is followed in the code by a pair of parentheses (). That is because it is a function declaration: In C++, what differentiates a function declaration from other types of expressions are these parentheses that follow its name. Optionally, these parentheses may enclose a list of parameters within them.

Right after these parentheses we can find the body of the main function enclosed in braces ({}). What is contained within these braces is what the function does when it is executed.

```
cout << "Hello World!";
```

This line is a C++ statement. A statement is a simple or compound expression that can actually produce some effect. In fact, this statement performs the only action that generates a visible effect in our first program.

cout represents the standard output stream in C++, and the meaning of the entire statement is to insert a sequence of characters (in this case the Hello World sequence of characters) into the standard output stream (which usually is the screen).

cout is declared in the iostream standard file within the std namespace, so that's why we needed to include that specific file and to declare that we were going to use this specific namespace earlier in our code.

Notice that the statement ends with a semicolon character (;). This character is used to mark the end of the statement and in fact it must be included at the end of all expression statements in all C++ programs (one of the most common syntax errors is indeed to forget to include some semicolon after a statement).

```
return 0;
```

The return statement causes the main function to finish. return may be followed by a return code (in our example is followed by the return code 0). A return code of 0 for the main function is generally interpreted as the program worked as expected without any errors during its execution. This is the most usual way to end a C++ console program.

You may have noticed that not all the lines of this program perform actions when the code is executed. There were lines containing only comments (those beginning by //). There were lines with directives for the compiler's preprocessor (those beginning by #). Then there were lines that began the declaration of a function (in this case, the main function) and, finally lines with statements (like the insertion into cout), which were all included within the block delimited by the braces ({} of the main function.

The program has been structured in different lines in order to be more readable, but in C++, we do not have strict rules on how to separate instructions in different lines. For example, instead of

```
int main ()  
{  
    cout << "Hello World!";  
    return 0;  
}
```

We could have written:

```
int main ()  
{  
    cout << "Hello World!";  
    return 0;  
}
```

All in just one line and this would have had exactly the same meaning as the previous code.

In C++, the separation between statements is specified with an ending semicolon (;) at the end of each one, so the separation in different code lines does not matter at all for this purpose. We can write many statements per line or write a single statement that takes many code lines. The division of

Cascading Of I/O Operator:

```
cout<<"sum="<<sum<<"\n";
cout<<"sum="<<sum<<"\n"<<"average="<<average<<"\n";
cin>>number1>>number2;
```

Structure Of A Program :

Probably the best way to start learning a programming language is by writing a program. Therefore, here is our first program:

```
// my first program in C++
```

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

```
int main ()
{
    cout << "Hello World!";
    return 0;
}
```

Output:-Hello World!

The first panel shows the source code for our first program. The second one shows the result of the program once compiled and executed. The way to edit and compile a program depends on the compiler you are using. Depending on whether it has a Development Interface or not and on its version. Consult the compilers section and the manual or help included with your compiler if you have doubts on how to compile a C++ console program.

The previous program is the typical program that programmer apprentices write for the first time, and its result is the printing on screen of the "Hello World!" sentence. It is one of the simplest programs that can be written in C++, but it already contains the fundamental components that every C++ program has. We are going to look line by line at the code we have just written:

```
// my first program in C++
```

This is a comment line. All lines beginning with two slash signs (*//*) are considered comments and do not have any effect on the behavior of the program. The programmer can use them to include short explanations or observations within the source code itself. In this case, the line is a brief description of what our program is.

```
#include <iostream>
```

Lines beginning with a hash sign (*#*) are directives for the preprocessor. They are not regular code lines with expressions but indications for the compiler's preprocessor. In this case the directive `#include<iostream>` tells the preprocessor to include the `iostream` standard file. This specific file (`iostream`) includes the declarations of the basic standard input-output library in C++, and it is included because its functionality is going to be used later in the program.

```
using namespace std;
```

All the elements of the standard C++ library are declared within what is called a namespace, the namespace with the name *std*. So in order to access its functionality we declare with this expression that we will be using these entities. This line is very frequent in C++ programs that use the standard library, and in fact it will be included in most of the source codes included in these tutorials.

```
int main ()
```

This line corresponds to the beginning of the definition of the main function. The main function is the point by where all C++ programs start their execution, independently of its location within the source code. It does not matter whether there are other functions with other names defined before or after it – the instructions contained within this function's definition will always be the first ones to be

LECTURE-5

Basics of C++

C ++ is an object oriented programming language, C ++ was developed by Jarney Stroustrup at AT & T Bell lab, USA in early eighties. C ++ was developed from c and simula 67 language. C ++ was early called 'C with classes'.

C++ Comments:

C++ introduces a new comment symbol //(double slash). Comments start with a double slash symbol and terminate at the end of line. A comment may start any where in the line and what ever follows till the end of line is ignored. Note that there is no closing symbol.

The double slash comment is basically a single line comment. Multi line comments can be written as follows:

```
// this is an example of  
// c++ program  
// thank you
```

The c comment symbols /* ...*/ are still valid and more suitable for multi line comments.

```
/* this is an example of c++ program */
```

Output Operator:

The statement `cout <<"Hello, world"` displayed the string with in quotes on the screen. The identifier `cout` can be used to display individual characters, strings and even numbers. It is a predefined object that corresponds to the standard output stream. Stream just refers to a flow of data and the standard Output stream normally flows to the screen display. The `cout` object, whose properties are defined in `iostream.h` represents that stream. The insertion operator `<<` also called the 'put to' operator directs the information on its right to the object on its left.

Return Statement:

In C++ `main ()` returns an integer type value to the operating system. Therefore every `main ()` in C++ should end with a `return (0)` statement, otherwise a warning or an error might occur.

Input Operator:

```
The statement  
cin>> number 1;
```

is an input statement and causes. The program to wait for the user to type in a number. The number keyed in is placed in the variable `number1`. The identifier `cin` is a predefined object in C++ that corresponds to the standard input stream. Here this stream represents the key board.

The operator `>>` is known as get from operator. It extracts value from the keyboard and assigns it to the variable on its right.

code in different lines serves only to make it more legible and schematic for the humans that may read it.

Let us add an additional instruction to our first program:

```
// my second program in C++
#include <iostream>
using namespace std;
```

```
int main ()
{
    cout << "Hello World! ";
    cout << "I'm a C++ program";
    return 0;
}
```

Output:-Hello World! I'm a C++ program

In this case, we performed two insertions into cout in two different statements. Once again, the separation in different lines of code has been done just to give greater readability to the program, since main could have been perfectly valid defined this way:

```
int main ()
{
    cout << " Hello World! ";
    cout << " I'm a C++ program ";
    return 0;
}
```

We were also free to divide the code into more lines if we considered it more convenient:

```
int main ()
{
    cout << "Hello World!";
    cout << "I'm a C++ program";
    return 0;
}
```

And the result would again have been exactly the same as in the previous examples.

Preprocessor directives (those that begin by #) are out of this general rule since they are not statements. They are lines read and processed by the preprocessor and do not produce any code by themselves. Preprocessor directives must be specified in their own line and do not have to end with a semicolon (;).

STRUCTURE OF C++ PROGRAM

- Include files
- Class declaration
- Class functions, definition
- Main function program

Example :-

```
# include<iostream.h>
```

```
class person
```