MVJ COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow

Approved by AICTE, New Delhi
Affiliated to VTU, Belagavi
Recognized by UGC under 2(f) &12(B)
Accredited by NBA & NAAC

Guest Lecture Organized by the Department of Computer Science and Engineering

Guest Lecture on Compiler Design

The Department of Computer Science & Engineering, MVJCE, organised a Guest Lecture on 'Compiler Design'. The Guest Speaker was Dr. Muralikrishnan K, Associate Professor, Department of Computer Science & Engineering, National Institute of Technology, NIT, Calicut, Kerala 673601.

The session was conducted at Seminar Hall 4, from 9.30 am to 12.40 pm, on 16th March, 2019.

Participants

The Lecture was attended by 48 students of the 6th Semester and 3 faculty members, all from the Department of Computer Science & Engineering.

About the Speaker

Dr. Muralikrishnan K is an Associate Professor in the Department of Computer Science and Engineering, National Institute of Technology, Calicut. He has been associated with the Department since 1998. His research publications are in the areas of low-density parity check codes, structure of planar graphs and program analysis. His general interests include complexity, algorithms and coding theory. He is also interested in the design of open source software tools for undergraduate Computer Science education, and has designed open source tools for instruction in operating systems and compilers, at undergraduate laboratories.

Dr. Muralikrishnan completed B. Tech in Computer Engineering with first rank, from Cochin University of Science and Technology, in 1996. He completed M. Tech in Computer Science and Engineering from IIT Kanpur in 1998, and Ph. D from Indian Institute of Science, Bangalore, in 2008. His Ph. D thesis, on the theory of error correcting codes, was awarded the Best Thesis of the Year, 2008, by the Department of Computer Science and Automation, IISc., Bangalore.

Dr. Muralikrishnan has also served as a member of the governing council of NIT Calicut, during 2011-2013.

A brief look at the Contents of the Lecture

The lecture started with a brief introduction of system software, and the differences between system software and application software. A Compiler was defined as a program that converts high-level language to a low-level language. Further, the following points that will set the stage for building a compiler for an experimental programming language to be run over a virtual machine, were elaborated:

1. **Virtual machine model:** The virtual machine model explains the 'view' of the machine provided to the application, by the operating system. Virtual machine contains 20 general purpose registers (R0-R19) and three special registers - Stack pointer (SP), Base pointer (BP) and the Instruction pointer (IP). A high-level picture of the XSM virtual machine model was also explained.

2. **Instruction set:** A brief introduction of data transfer instructions, arithmetic instructions, logical instructions, branching instructions, stack instructions, subroutine instructions, debug instructions, and system call with respect to interrupt routine number, was given.

3. **The (virtual) address space model**: This is available for the target program. The address space is logically divided into regions like library, heap, code, stack etc.

    a)The **Library** contains the routines for dynamic memory allocation, and input and output functions between the addresses 0 and 1023 of the address space.

    b)For dynamic memory allocation, **Heap** address space is reserved by the allocator routine of the library. The memory region between addresses 1024 and 2047 is reserved for the heap.

    c)The **Code** region contains the target assembly language program generated by the compiler, between memory addresses 2048 and 4095 of the address space.

    d)**Stack** is the space reserved for the runtime stack of a program, between memory addresses 4095 and 5119 in the address space.

4. **Executable format:** The compiler typically passes information regarding the sizes and address regions allocated to the code, data, stack and heap regions to the loader, by setting appropriate values in the header of the executable file.

5. **Interfaces to OS (kernel) routines:** A compiler needs to be concerned to translate high-level function like dynamic memory allocation / de-allocation, and input and output functions.

By using all the above information, the compiler code generation and abstract syntax tree formation was taught, for basic programs like print, addition of two numbers and if-then-else structure, for evaluation of an arithmetic expression.

Guest Lecture on "Compiler Design" organized by Department of Computer Science & Engineering: The Resource Person Dr. Muralikrishnan K., Associate Professor, Department of CSE, NIT Calicut is delivering a lecture on building a compiler at Seminar Hall No.4 on 16th March, 2019.

Guest Lecture on "Compiler Design" organized by Department of Computer Science & Engineering: The student participants from 6th sem CSE and faculty members from the department of CSE during the session at Seminar hall No.4 on 16th March, 2019.

Outcome of the Event:

The lecture will certainly enable students to implement their own compiler for a simple procedural language, using a self-learning platform.