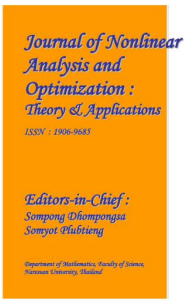
**J**ournal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 15, Issue. 1, No.15 : 2024
ISSN : **1906-9685**




**Paper ID:** ICRTEM24_101 **ICRTEM-2024 Conference Paper**

# ONLINE AUTONOMOUS LEARNING OF UNIVERSITY PROGRAMMING COURSES BY UNDERGRADUATE STUDENTS

**[1]N.Srihari Rao, *Professor*,**
**Bharat Institute of Engineering and Technology, Hyderabad,**
**[2]G. Deepika, *Assistant Professor*, CVR College of Engineering, Hyderabad**
**[3]K.Bhaskar, [4]S.Joshna, [5]B.Ramya Sree, [6]U.Narsimhulu**
**[#3,4,5,6]*Assistant Professor***
**Bharat Institute of Engineering and Technology, Hyderabad**

**Abstract:** In general, autonomous learning behavior refers to an individual's ability to take charge of their own learning process, independently seeking and acquiring knowledge. In this paper, we explored most of the important ways to encourage and enhance online autonomous learning behavior for any undergraduate computer science student. Analyzing students' learning attitudes stands crucial for understanding their approach to education and identifying areas that may need improvement. Programming language concepts and the related skills are very much important to the people who want to become software product developers. Programming languages stand as the most important component without which one can't be appointed for the software development tasks by the companies. Online autonomous learning of programming languages involves individuals taking control of their own learning process, leveraging online resources, and tools to acquire programming skills independently. Our methodology proposed in this paper consists of analyzing students' learning attitudes first as a step towards encouraging online learning and to foster online autonomous learning behavior of students towards programming languages. Coding using one or more programming languages to solve a specific problem by the people is the ultimate, direct and important outcome. Online automatic judgment of coding suggests the use of the automated tools or systems to evaluate and assess code written by individuals. We considered most of the important aspects related to online automatic judgment of coding by different people. Finally, we highlighted the important benefits of online autonomous learning.

**Keywords:** Online learning, autonomous learning, learning behavior/attitude, university programming course, coding skills.

# I. Introduction

Autonomous learning behavior refers to an individual's ability to take charge of their own learning process, independently seeking and acquiring knowledge. Autonomous learning behavior in an online context involves taking control of one's own learning process, leveraging digital resources and tools for self-directed education. In an online context, where a plethora of resources are available, fostering autonomous learning behavior is crucial. Following are some general ways to encourage and enhance online autonomous learning behavior among students.

**Set Clear Goals:** Define specific, measurable, achievable, relevant, and time-bound (SMART) learning goals. Break down larger goals into smaller, manageable tasks.

**Create a Learning Plan:** Develop a structured plan outlining what to learn, how to learn it, and the resources required. Allocate time for regular reviews and adjustments to the plan.

**Resource Selection:** Explore a variety of online resources such as courses, articles, videos, and interactive platforms. Choose resources that align with your learning style and preferences.

**Utilize Online Platforms:** Explore a variety of online learning platforms, such as Coursera, edX, Khan Academy, and others. Choose courses and resources that align with personal interests and career goals.

**Effective Time Management:** Develop a schedule that balances learning activities with other commitments. Prioritize tasks based on their importance and deadlines.

**Active Engagement:** Actively participate in online forums, discussion groups, and social media communities related to the chosen topic. Engage in discussions, ask questions, and share knowledge with others.

**Continuous Reflection:** Reflect on learning progress regularly, identifying strengths, weaknesses, and areas for improvement. Adjust learning strategies based on reflection outcomes. Regularly assess your understanding through quizzes, tests, or self-assessment exercises.

**Use of Multiple Resources:** Explore diverse learning materials, such as articles, videos, podcasts, and interactive simulations. Cross-reference information from different sources to gain a comprehensive understanding.

**Practice Self-Assessment:** Regularly assess your understanding of the material through self-assessment quizzes, projects, or practice exercises. Identify areas that need further attention and focus additional efforts there.

**Build a Personal Learning Network (PLN):** Connect with professionals, experts, and peers in the chosen field through social media, forums, and online communities. Seek mentorship or guidance from experienced individuals. Participate in online communities to share knowledge and insights.

**Embrace Challenges:** View challenges and setbacks as opportunities for growth. Push beyond the comfort zone to tackle more complex topics and skills.

**Utilize Technology Tools:** Leverage productivity and organizational tools such as note-taking apps, task management apps, and collaborative platforms.

**Stay Informed:** Keep up with industry trends, advancements, and emerging topics relevant to the chosen field. Subscribe to newsletters, follow blogs, and attend webinars or online conferences. Cultivate a mindset of continuous learning by staying curious and open to new ideas. Explore emerging trends and advancements in your field.

**Celebrate Achievements:** Acknowledge and celebrate milestones and achievements, providing motivation to continue the learning journey.

**Self-Motivation:** Develop intrinsic motivation by understanding the personal relevance and importance of the subject matter. Set long-term and short-term goals to maintain focus and drive.

**Critical Thinking:** Develop critical thinking skills by questioning information, analyzing different perspectives, and evaluating evidence. Encourage a curious and questioning mindset.
**Feedback Seeking:** Actively seek feedback on your work and understanding. Use feedback as a tool for improvement and refinement.
**Adaptability:** Embrace change and be adaptable to new technologies, methodologies, and learning environments. Continuously reassess and adjust your learning strategies based on feedback and experience.
**Technology Proficiency:** Develop proficiency in using online tools, learning management systems, and other digital resources. Stay updated on technological advancements relevant to your field.
**Project-Based Learning:** Engage in project-based learning to apply theoretical knowledge in practical scenarios. Develop and complete projects that align with your learning goals.
**Self-Regulation:** Take responsibility for your learning process, making decisions about what to learn, how to learn, and when to seek help.
By incorporating the above strategies, individuals can cultivate a proactive approach to learning and thrive in the ever-evolving landscape of online education. This way, individuals can develop a strong sense of autonomy in their online learning endeavors, allowing for a more personalized and effective educational experience.

This paper is organized as follows: Section II presents the related work. Section III contains the proposed methodology as a series of steps for online autonomous learning of university programming courses by undergraduate students. Section IV summarizes the benefits of online autonomous learning. Section V concludes the paper with the mention of future scope.

## II. Related Work

Xue Bai et al. [1] aim to study the autonomous learning behavior of college students in blended learning. In their study, five hypotheses were answered by using multiple linear regression method. They showed that learning motivation and academic self-efficacy can effectively predict college students' autonomous learning behavior. Mo Zhihui [2] investigates the current situation and problems of college students' learning behaviors under the mobile learning environment by using a combination of questionnaire survey and interview with undergraduates in a certain university in a certain region as the research object. Shuang Li et al. [3] investigated the current status of learners' online learning behaviors with different grades and explore the relationship between learning behaviors and Self-regulated learning levels. The study found that the high score group generally have higher learning engagement and are more likely to be influenced by goal setting. Juanjuan Cheng [4] researched and designed the intelligent autonomous learning management system to meet the needs of the current education field for the intelligent autonomous learning management system.
Yuting Zhao et al. [5] researched the status of precision teaching at home and abroad, studying the development path of current information teaching, analyzing the problems and influencing factors in the current precision teaching. Y. Zhang et al. [6] proposed a classroom behavior detection method for students based on the Human-Object Interaction (HOI) model, which further utilizes human-object relationship features to infer interaction relationships based on object detection. H. Zhu et al. [7] proposed that AI and big data analysis for student classroom behavior recognition can be used as auxiliary means to improve teaching quality.

## III. Proposed Method

Our methodology consists of analyzing students' learning attitudes first as a step towards encouraging online learning and to foster online autonomous learning behavior of students towards programming languages. The faculty engages students in a variety of activities in order to promote learning of programming courses and conducts quizzes in order to understand the real difficult areas of students while learning. The online automatic judgment of coding will enable the tutors to easily evaluate the coding skills of a large group of students and take necessary course of action towards the backward students for improvement.

### A. Analyzing Students' Learning Attitudes

Analyzing students' learning attitudes is crucial for understanding their approach to education and identifying areas that may need improvement. Learning attitudes encompass a range of characteristics, including motivation, engagement, perseverance, and beliefs about learning.

**How to Analyze Students' Learning Attitudes?**
Below are mentioned some key factors to consider when analyzing students' learning attitudes.

**Motivation:** Assess students' intrinsic and extrinsic motivation levels. Look for signs of enthusiasm, interest in the subject matter, and a desire to achieve academic success.

**Interest and Curiosity:** Observe students' level of interest in the material. Identify whether students show curiosity, ask questions, and seek additional information beyond the required curriculum.

**Perseverance and Resilience:** Evaluate how students respond to challenges and setbacks. Identify whether students exhibit resilience, perseverance, and a willingness to overcome obstacles in their learning journey.

**Responsibility and Accountability:** Assess students' sense of responsibility for their own learning. Determine whether students take ownership of their academic performance and actively engage in the learning process.

**Goal Orientation:** Identify whether students set and work towards specific learning goals. Differentiate between performance-oriented (focused on grades) and mastery-oriented (focused on understanding) goal orientations.

**Self-Regulation:** Observe students' ability to manage their time effectively. Evaluate whether students can regulate their learning behaviors, such as setting priorities, staying organized, and avoiding procrastination.

**Collaboration and Communication:** Assess how students interact with peers and collaborate on group projects. Identify their communication skills, including the ability to express ideas, ask questions, and provide constructive feedback.

**Metacognition:** Evaluate students' awareness of their own learning processes. Determine whether students can reflect on their understanding, identify areas of difficulty, and employ effective learning strategies.

**Attitudes Towards Feedback:** Observe how students respond to feedback from teachers and peers. Identify whether they view feedback as a tool for improvement and demonstrate a willingness to make necessary adjustments.

**Engagement in Class Activities:** Monitor students' participation in class discussions, activities, and assignments. Evaluate their level of engagement and active involvement in the learning process.

**Attitudes Towards Collaboration:** Assess how students collaborate with their peers in group activities. Identify whether they value teamwork, appreciate diverse perspectives, and contribute positively to collaborative efforts.

**Interest in Continuous Learning:** Gauge students' interest in ongoing learning beyond the classroom. Identify whether they pursue additional resources, attend relevant events, or

engage in extracurricular activities related to their field of study.

Analyzing students' learning attitudes involves a combination of observation, discussions, and assessments. Regular communication with students, feedback sessions, and surveys can provide valuable insights into their attitudes towards learning, allowing educators to tailor their instructional approaches to better meet individual needs.

### B. What does Online Autonomous Learning of University Programming Courses Involve?

Online autonomous learning of University programming courses [10] involve individuals taking control of their own learning process, leveraging online resources and tools to acquire programming skills independently. Below are given key behaviors and strategies associated with autonomous learning of programming languages online.

**Goal Setting:** Set clear and achievable learning goals for acquiring programming skills. Break down larger goals into smaller, manageable tasks or projects.

**Resource Selection:** Explore a variety of online resources, including tutorials, documentation, video lectures, and interactive coding platforms. Choose resources that cater to your preferred learning style and the programming language you want to learn.

**Structured Learning Plan:** Develop a structured learning plan outlining the sequence of topics to cover. Allocate specific time slots for learning and practice within your schedule.

**Hands-On Coding Practice:** Actively engage in hands-on coding practice to reinforce theoretical knowledge. Work on coding exercises, challenges, and small projects to apply what you've learned.

**Project-Based Learning:** Undertake projects that align with your interests or solve real-world problems. Apply programming concepts to build practical solutions, gaining valuable experience.

**Online Courses and Platforms:** Enroll in online programming courses on platforms like Codecademy, Coursera, edX, or Udacity. Follow structured courses that cover fundamentals and advanced topics in the chosen programming language.

**Community Engagement:** Participate in online programming communities and forums. Ask questions, seek advice, and collaborate with other learners and experienced developers.

**Documentation and Reference Usage:** Learn to navigate and use programming language documentation effectively. Refer to official documentation and online resources to deepen your understanding of language features and best practices.

**Version Control Systems:** Familiarize yourself with version control systems, such as Git. Use version control for managing your code, collaborating with others, and tracking changes.

**Code Review and Feedback:** Seek feedback on your code by sharing it with peers or participating in online code review platforms. Learn from constructive feedback and continuously improve your coding practices.

**Continuous Learning:** Stay updated on new language features, best practices, and industry trends. Subscribe to newsletters, follow blogs, and engage in discussions to remain informed.

**Problem-Solving Skills:** Develop problem-solving skills by tackling coding challenges on platforms like HackerRank, LeetCode, or Codewars. Approach problem-solving systematically and practice algorithmic thinking.

**Networking and Mentoring:** Network with experienced developers through social media, LinkedIn, or local meetups. Seek mentorship to gain insights, guidance, and advice from seasoned professionals.

**Self-Assessment:** Regularly assess your understanding through quizzes, coding assessments, or personal projects. Identify areas

for improvement and focus on enhancing your weaknesses.

**Portfolio Development:** Build a portfolio showcasing your projects and achievements. Share your portfolio on platforms like GitHub to demonstrate your coding skills to potential employers.

Autonomous learning of programming languages online is a dynamic and ongoing process that requires dedication, curiosity, and a proactive approach. By adopting these behaviors and strategies, individuals can effectively progress in their programming language learning journey.

### C. How to Foster Online Autonomous Learning of Programming Languages?

To foster novel techniques for online autonomous learning of programming languages, it's important to leverage emerging technologies and innovative strategies [10]. Given below are few novel approaches and techniques.

**Interactive Coding Environments:** Explore interactive online coding environments that provide a seamless learning experience. Platforms like Repl.it, Glitch, or Jupyter Notebooks allow learners to code directly in the browser with instant feedback.

**Gamification:** Integrate gamification elements into programming courses to make learning more engaging. Platforms like CodeCombat or CheckiO use game-like scenarios to teach programming concepts.

**Augmented Reality (AR) and Virtual Reality (VR):** Experiment with AR and VR applications to create immersive coding environments. Virtual coding environments can enhance the hands-on learning experience for programming languages.

**AI-Powered Tutors:** Implement AI-driven tutoring systems that adapt to individual learning styles. These systems can provide personalized feedback, suggest tailored exercises, and adapt the learning pace based on the user's progress.

**Project-Based Learning Platforms:** Utilize platforms that focus on project-based learning, allowing learners to work on real-world projects collaboratively. GitHub Classroom and GitLab are examples of platforms that support project-based learning.

**Natural Language Processing (NLP) for Coding Assistance:** Integrate natural language processing capabilities into coding environments to assist learners in expressing coding concepts in plain language. AI tools like OpenAI's Codex can help generate code snippets based on natural language queries.

**Blockchain-Based Credentialing:** Explore blockchain-based credentialing systems to provide verifiable proof of skills acquisition. Blockchain can be used to issue digital badges or certificates for completing coding challenges or projects.

**Interactive Learning Videos:** Develop interactive video content that allows learners to code directly within the video interface. Tools like Kaltura or H5P can enhance traditional video content with interactive coding elements.

**Microlearning Modules:** Break down programming concepts into bite-sized microlearning modules. Platforms like SoloLearn offer mobile-friendly microlearning experiences, allowing learners to engage in short coding exercises on the go.

**Personalized Learning Paths:** Implement adaptive learning algorithms that dynamically adjust the learning path based on individual progress and performance. Personalization engines like Knewton or DreamBox Learning can be adapted for programming education.

**Social Coding Platforms:** Leverage social coding platforms that encourage collaboration and peer-to-peer learning. GitHub and GitLab provide opportunities for learners to contribute to open-source projects and collaborate with others.

**Code Visualization Tools:** Integrate tools that visually represent code execution flow and data

structures. Visualizations can enhance understanding and make complex programming concepts more accessible.

**Cloud-Based Development Environments:** Utilize cloud-based IDEs and development environments that allow learners to access their projects from anywhere. Cloud9 and Visual Studio Online are examples of cloud-based IDEs.

**AI-Driven Code Reviews:** Implement AI-powered code review tools that analyze code quality, suggest improvements, and provide detailed feedback. These tools can emulate the insights of experienced developers in real-time.

**Voice-Activated Coding:** Experiment with voice-activated coding interfaces, allowing learners to verbally dictate code. This can be beneficial for individuals with different learning preferences and accessibility needs.

By integrating these novel techniques, online autonomous learning of programming languages can become more dynamic, engaging, and tailored to individual learner needs. As technology continues to evolve, exploring and incorporating innovative approaches will contribute to a richer and more effective learning experience.

### D. Online Automatic Judgment of Codings

The concept of "online automatic judgment of codings" suggests the use of automated tools or systems to evaluate and assess code written by individuals. This process is often referred to as code analysis, code review, or automated code assessment. Given below are some significant aspects related to the online automatic judgment of codings.

**Code Quality Metrics:** Automated tools can analyze code quality metrics, including readability, maintainability, and adherence to coding standards. Metrics such as cyclomatic complexity, code duplication, and code smells can be assessed automatically.

**Syntax and Style Checking:** Tools can perform syntax checking to ensure that the code adheres to the rules of the programming language. Style checking can enforce consistent coding styles and formatting conventions.

**Security Vulnerability Detection:** Automated tools can identify potential security vulnerabilities in the code. They may detect common issues like SQL injection, cross-site scripting (XSS), or insecure dependencies.

**Performance Analysis:** Tools can analyze code for performance bottlenecks or inefficient constructs. They may provide suggestions for optimizing code to improve runtime performance.

**Code Documentation Assessment:** Automated systems can assess the quality of code documentation. This includes checking for the presence of comments, documentation comments, and the use of meaningful variable/method names.

**Testing and Test Coverage:** Some tools assess the quality of test code and check for sufficient test coverage. They may identify areas of code that lack proper testing.

**Dependency Analysis:** Automated tools can analyze code dependencies, checking for outdated or vulnerable third-party libraries. They may suggest updates to dependencies to address security concerns.

**Code Review Automation:** Automated code review tools can simulate human code reviews. They may provide feedback on code changes, identifying potential issues and suggesting improvements.

**Machine Learning-based Code Analysis:** Some advanced tools leverage machine learning to identify patterns and anomalies in code. They can learn from large codebases to provide more context-aware suggestions.

**Integration with Version Control Systems:** Tools can integrate with version control systems to analyze code changes as part of the continuous integration/continuous deployment

(CI/CD) pipeline. They may prevent the integration of code that doesn't meet predefined criteria.

**Plagiarism Detection:** Automated tools can detect instances of code plagiarism or code similarity. This is especially useful in educational settings to ensure code integrity.

**Feedback and Reporting:** Automated systems provide feedback to developers, often in the form of reports or annotations within integrated development environments (IDEs). They may categorize issues based on severity and provide recommendations for improvement.

It's important to note that while automated tools can significantly enhance the code quality assessment process, they may not capture all aspects of code quality. A combination of automated tools and manual code reviews by experienced developers is often the most effective approach to ensure high-quality code.

## III. Benefits of Online Autonomous Learning

Given below are the few important benefits of online autonomous learning behavior.

**Flexibility:** Learners can choose their pace and schedule, adapting the learning process to their lifestyle.

**Access to Diverse Resources:** Online platforms offer a wide range of resources, catering to various learning styles.

**Cost-Effective:** Many online resources are either free or more affordable than traditional education.

**Global Collaboration:** Learners can connect with others worldwide, fostering collaboration and diverse perspectives.

**Real-world Application:** Project-based learning and coding challenges provide practical, job-relevant skills.

## IV. Conclusion and Future Scope

It is vital to encourage and enhance online autonomous learning behavior for any computer science subject student. Analyzing students' learning attitudes stands tall for understanding their approach to education and identifying areas that may need improvement. Online autonomous learning of programming languages involves individuals taking control of their own learning process, leveraging online resources and tools to acquire programming skills independently. We explored the key behaviors and strategies associated with autonomous learning of programming languages online. In conclusion, the online autonomous learning of programming languages involves a dynamic combination of interactive platforms, documentation, coding challenges, communities, and continuous practice. The flexibility and accessibility of online resources empower individuals to shape their learning journey and stay current in the ever-evolving field of programming. The result of this study will be utilized to understand the big picture and perform a case study by using Precision Teaching Classroom (PTC) methodology [8-9] for University Programming courses.

## References

[1] X. Bai, X. Wang, J. Wang, J. Tian and Q. Ding, "College Students' Autonomous Learning Behavior in Blended Learning: Learning Motivation, Self-Efficacy, and Learning Anxiety," 2020 International Symposium on Educational Technology (ISET), Bangkok, Thailand, 2020, pp. 155-158, doi: 10.1109/ISET49818.2020.00042.

[2] M. Zhihui, "Research on Influencing Factors and Optimization of College Students' Learning Behavior Based on Mobile Learning," 2021 2nd International Conference on Big Data and Informatization Education (ICBDIE), Hangzhou, China, 2021, pp. 483-487, doi: 10.1109/ICBDIE52740.2021.00116.

[3] S. Li, J. Q. Sun, Z. x. Huang and J. Q. Chen, "The Relationship between Self-regulated Learning and Learning Behaviors of Chinese Middle School Students in a Smart Classroom," 2021 Tenth International Conference of Educational Innovation through Technology (EITT), Chongqing, China, 2021, pp. 350-355, doi: 10.1109/EITT53287.2021.00075.

[4] J. Cheng, "Design of Intelligent Autonomous Learning Management System Based on Artificial Intelligence Technology," 2022 International Conference on Education, Network and Information Technology (ICENIT), Liverpool, United Kingdom, 2022, pp. 106-110, doi: 10.1109/ICENIT57306.2022.00030.

[5] Y. Zhao, Y. Lei, M. Li and D. Xin, "Construction of Accurate Teaching Model Based on Intelligent Teaching Tools—Take the Rain Classroom As An Example," 2020 International Conference on Big Data and Informatization Education (ICBDIE), Zhangjiajie, China, 2020, pp. 317-321, doi: 10.1109/ICBDIE50010.2020.00080.
[6] Y. Zhang, W. Qu, G. Zhong and Y. Xiao, "Students' Classroom Behavior Detection Based on Human-Object Interaction Model," 2022 8th International Conference on Systems and Informatics (ICSAI), Kunming, China, 2022, pp. 1-6, doi: 10.1109/ICSAI57119.2022.10005457.
[7] H. Zhu, J. Zhao and L. Niu, "An Efficient Model For Student Behavior Recognition in Classroom," 2022 International Conference on Intelligent Education and Intelligent Research (IEIR), Wuhan, China, 2022, pp. 142-147, doi: 10.1109/IEIR56323.2022.10050077.
[8] F. Yu, Y. Liu and F. Xiao, "Research on Construction and Practice of Precision Teaching Classroom for University Programming Courses," in IEEE Access, vol. 11, pp. 9560-9576, 2023, doi: 10.1109/ACCESS.2023.3240105.
[9] S. Ma, "The Construction and Practice of Precision Teaching Model of "Smart Class" based on "Internet + Education"," 2021 3rd International Conference on Internet Technology and Educational Informization (ITEI), Guangzhou, China, 2021, pp. 1-4, doi: 10.1109/ITEI55021.2021.00009.
[10] ChatGPT 3.5, URL: https://chat.openai.com/