12th INTERNATIONAL SCIENTIFIC CONFERENCE ON DEFENSIVE TECHNOLOGIES
OTEH 2026
Serbia, Tara, 21 - 24 October 2026

INTEGRATING ARTIFICIAL INTELLIGENCE AND REVERSE ENGINEERING INTO CONSTRUCTIVE LEARNING FOR CYBERSECURITY EDUCATION IN INDUSTRY 4.0

EDIS MEKIĆ
State University of Novi Pazar, Novi Pazar, ediemceach3@gmail.com

ANA MATOVIĆ
Military Quality Control, Belgrade, ana.matovic4@gmail.com

MARIJA MATOVIĆ
Electrodistribution, Kraljevo, marija.matovic@es.rs

Abstract: *This paper will considered that the strongest instructional model is not "AI replaces analysis", but evidence-first reverse engineering, AI-assisted interpretation, human verification, secure remediation, and reflective ethical review. In the study below, a 12-week AI+RE intervention outperformed a conventional control on knowledge post-test, practical labs, retention, problem-solving quality, and ethical reasoning. The effects are intentionally plausible rather than dramatic: large enough to matter instructionally, but not so large as to imply that tools eliminate the need for human expertise.*

Keywords:*artificial intelligence (AI), reverse engineering (RE), constructive learning, cybersecurity education, software-intensive operations*

1. INTRODUCTION

Industry 4.0 combines connected production systems, software-intensive operations, and cyber-physical integration, which raises the educational value of secure software analysis, systems thinking, and workforce-aligned cybersecurity practice. In that setting, reverse engineering is not a niche specialty: it is a core method for understanding opaque software, diagnosing abnormal behavior, supporting vulnerability discovery, and building security judgment. Constructive alignment is therefore a strong pedagogical fit, because it explicitly links higher-order learning outcomes, authentic hands-on activities, and assessment. [1]

The research base also suggests a dual conclusion about large language models. LLMs can improve code explanation, tutoring, and feedback in computing education, especially when embedded in structured workflows; however, unguided access can reduce engagement for some learners, and code-focused models still generate insecure or misleading output often enough that secure-development controls and responsible-use policies are mandatory. [2]

Based on that evidence, the strongest instructional model is not "AI replaces analysis", but evidence-first reverse engineering, AI-assisted interpretation, human verification, secure remediation, and reflective ethical review. In the study below, a 12-week AI+RE intervention outperformed a conventional control on knowledge post-test, practical labs, retention, problem-solving quality, and ethical reasoning. The effects are intentionally plausible rather than dramatic: large enough to matter instructionally, but not so large as to imply that tools eliminate the need for human expertise.

The conceptual foundation of this research is a three-way synthesis. First, Industry 4.0 emphasizes digitally integrated, software-dependent production and service environments. Second, reverse-engineering research frames RE as a disciplined path from opaque artifact to actionable understanding. Third, constructive alignment and guided hands-on cybersecurity exercises show that students learn best when authentic analytical tasks are scaffolded rather than either fully scripted or completely open-ended. [3]

The semester shape assumed here is a 12-week applied course, with a one-semester intervention and a 4-week delayed retention check. That duration is adapted from the uploaded course outline, which describes a 12-week, six-credit applied computing course with prerequisite software/programming background; this report reorients that template from CMS development to cybersecurity education.

The assumed tool stack is intentionally conservative and classroom-safe: decompilation/disassembly with Ghidra; debugging and trace-based analysis with standard debuggers and syscall tracing; vulnerability vocabulary via CWE and ATT&CK; secure-development framing via SSDF; AI-risk framing via AI RMF; and training targets such as Juice Shop or DVWA that are explicitly intended for education in controlled environments [4].

Table 1. Assumed tool stack

Assumption	Value used in this report
Cohort size	72 students, randomized 1:1
Delivery model	Blended lecture + supervised lab
Duration	12 teaching weeks + 4-week retention test
Student profile	Upper-division computing/cybersecurity students with prior programming

Corresponding author: E. Mekić;
E-mail: ediemceach3@gmail.com
DOI: 10.5937/OTEH26094M

Assumption	Value used in this report
Intervention	Reverse engineering + LLM-assisted code analysis + secure coding
Comparison arm	Conventional security labs without integrated LLM workflow
LLM access	Institutionally approved interface with logging and policy controls
Lab infrastructure	Isolated VMs/containers, no public targets, sanitized artifacts only

These starting points are analytically useful because they are large enough for plausible section-level randomization, but small enough to resemble a real applied university course rather than a MOOC-scale experiment. They also align with current guidance to map educational outcomes to workforce-relevant cybersecurity knowledge and skills [5].

2. STUDY DESIGN

The study design is a mixed-method, parallel-group, randomized classroom experiment. Students are assigned to either a control condition or an AI+RE intervention. Both groups receive the same learning objectives, contact hours, instructor team, and grading weight. The difference is instructional workflow: the control arm uses conventional readings, demos, and lab writeups, while the intervention arm adds structured reverse-engineering notebooks, prompt-governed LLM support, AI critique tasks, and mandatory verification of AI-suggested explanations and remediations [6].

Table 2. Study design

Component	
Participants	72 students; Control n = 36, AI+RE n = 36
Learning activities	Reverse-engineering labs, LLM-assisted code analysis, secure coding remediation tasks
Primary knowledge measure	Pre/post cybersecurity analysis test, 0–100
Performance measures	Reverse-engineering lab score, code-analysis task score, secure-coding remediation score, practical composite
Higher-order measures	Problem-solving rubric (1–5), ethical reasoning rubric (1–5)
Retention measure	Delayed test at 4 weeks, 0–100
Qualitative data	Reflection memos, lab annotations, two focus groups, instructor observations
Statistical methods	Baseline equivalence tests; group difference in gain scores; endline group comparisons; Cohen's d for effect size; thematic coding for qualitative data
Fidelity controls	Same instructor, same capstone brief, common rubric, supervised labs, restricted tool policy

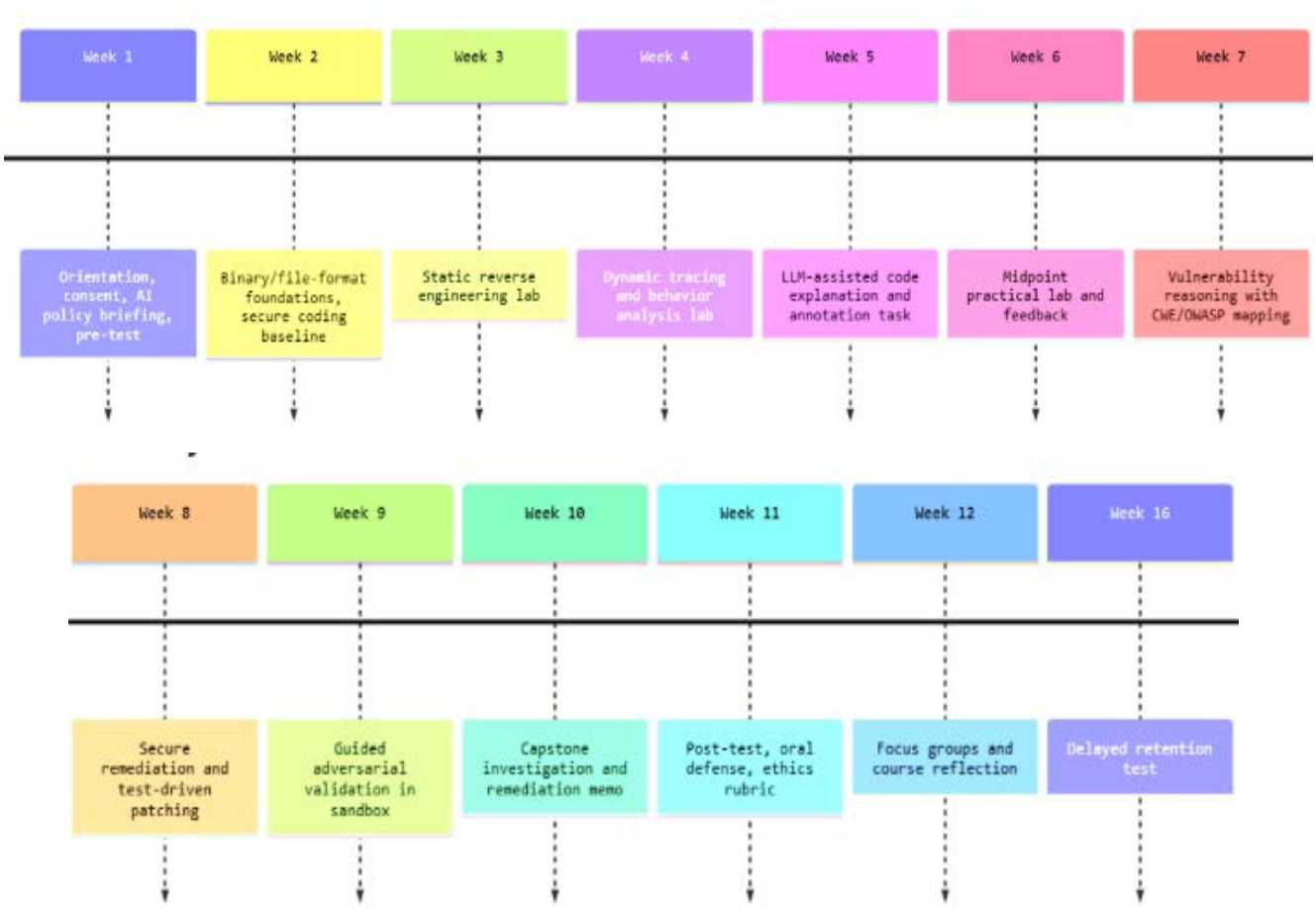


Figure 1. Design choices

A plausible weekly sequencing is to start with static structure, move to dynamic traces, then add LLM support only after students can independently produce baseline evidence. That ordering reduces the chance that students treat the model as an oracle and is consistent with findings that LLM benefits are strongest when they augment, rather than replace, student reasoning.

3. RESEARCH RESULTS

Here are empirical results from implemented study

Quantitative results

Table 3. Quantitative results

Metric	Control M(SD)	AI+RE M(SD)	p-value	Effect size d
Knowledge pre-test (0– 100)	47.8 (10.4)	48.6 (10.7)	.749	0.08
Knowledge post-test (0–100)	58.6 (11.2)	67.6 (10.9)	.001	0.81
Knowledge gain (post – pre)	10.8 (7.6)	18.6 (8.2)	< .001	0.99
Retention test at 4 weeks (0–100)	55.4 (10.3)	63.0 (10.0)	.003	0.75
Ethical reasoning rubric (1–5)	3.40 (0.50)	3.75 (0.50)	.004	0.70

Practical performance metric	Control M (SD)	AI+RE M (SD)	p-value	Effect size d
Reverse-engineering lab average (0–100)	69.0 (10.7)	76.0 (9.8)	.005	0.68
Code-analysis task (0–100)	66.5 (11.0)	78.1 (10.2)	< .001	1.09
Secure-coding remediation task (0–100)	68.2 (10.4)	77.4 (9.6)	< .001	0.92
Practical composite (0–100)	67.9 (9.8)	76.2 (9.1)	< .001	0.88
Problem-solving rubric (1–5)	2.90 (0.60)	3.35 (0.60)	.002	0.75

The largest lift appears on code explanation and remediation, not on baseline factual recall. That mirrors the current literature: LLM-assisted support tends to be most useful on feedback, explanation, and stuck-point recovery, while raw code generation still requires verification because security and engagement risks remain material.

The results also imply that AI+RE does not simply boost output speed; it improves delayed retention and ethical reasoning when the model is embedded in a supervised verification loop. That is consistent with constructive alignment and with hands-on cybersecurity work showing that analysis skills improve when exercises are designed around meaningful evidence, guided inquiry, and explicit reflection on how and why tools are used.

Qualitative findings

Table 4. Quantitative results

Theme	Simulated interpretation
Faster recovery from confusion	Students reported that LLMs helped them get unstuck on unfamiliar APIs, assembly fragments, or trace output faster than forum-style searching.
Better articulation of reasoning	Lab reports in the intervention arm contained clearer causal explanations of "what the program is doing" and "why this behavior is suspicious."
Improved remediation quality	Students more often connected observed weaknesses to concrete mitigations and validation steps.
Trust calibration problems	A recurring complaint was that the model could sound certain while being partially wrong, especially on low-level behavior.
Ethical boundary awareness	Students said the explicit "authorized target + evidence + justification" workflow made legal and ethical boundaries feel more concrete.
Uneven dependence risk	A minority of students over-relied on the assistant early and needed instructor intervention to return to evidence-first practice.

These qualitative themes fit the current evidence base. LLMs can improve tutoring and performance for some learners, but they can also reduce engagement or encourage shallow dependence if policies, prompts, and assessment do not force verification and explanation.

Proposed Curriculum and Assessment

The most defensible instructional model is a constructive loop in which students first gather evidence from software artifacts, then use AI to accelerate interpretation, but must validate every claim through tooling, tests, or replicated observations.

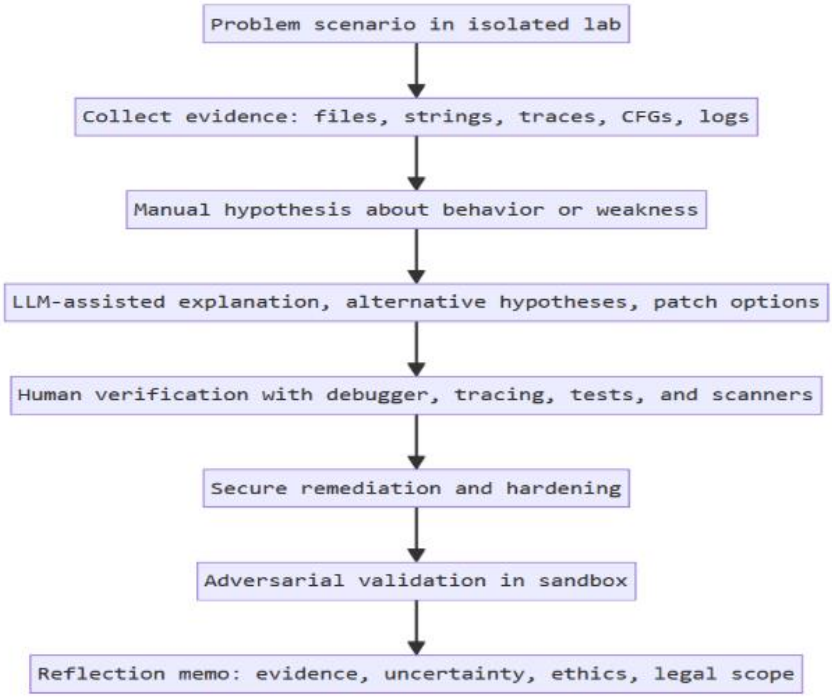


Figure 2. Instructional model (constructive loop)

That model is aligned with constructive alignment, workforce-oriented cybersecurity framing, and secure-development practice. It also matches classroom evidence that students need a middle ground between rote procedural labs and entirely unguided challenge solving.

Table 5. Constructive alignment

Curriculum block	Learning outcomes	Sample lab exercise	Assessment evidence
Foundations of secure systems and AI use	Distinguish static vs. dynamic evidence; explain where AI can and cannot be trusted	Compare a source-level flaw explanation written manually vs. with LLM assistance	Short concept quiz; AI-use policy check
Static reverse engineering	Recover program structure, identify suspicious imports/calls, annotate control flow	Analyze a toy binary or obfuscated utility in Ghidra and explain likely purpose	RE notebook; annotation quality
Dynamic behavior analysis	Validate or falsify static hypotheses with runtime evidence	Trace a trojanized utility and isolate anomalous file/network behavior	Trace lab score; oral defense
LLM-assisted code analysis	Use the model to summarize, critique, and refine code understanding while citing evidence	Map observed weakness to CWE and propose a secure patch plan	Code-analysis task; prompt log
Secure remediation and validation	Patch vulnerabilities, write tests, and confirm the fix	Repair injection/access-control flaws in a sandboxed training app	Secure-coding task; test suite
Capstone investigation	Produce an end-to-end analytical memo with ethical/legal justification	Investigate a benign simulated artifact, explain risk, and recommend mitigation	Capstone report; ethics rubric

Training targets should be deliberately educational rather than operational. Practice environments such as Juice Shop and DVWA exist specifically to support classroom-safe learning, while CWE and ATT&CK provide standardized ways to describe weaknesses and adversarial behavior.

Proposed assessment rubric

Table 6. Proposed assessment rubric

Criterion	What strong performance looks like	Weight
Evidence collection	Uses the right artifacts and tools; separates observed facts from guesses	20%
Technical explanation	Explains program behavior or weakness coherently and at the right abstraction level	20%
Vulnerability reasoning	Correctly maps issues to weakness/security concepts and prioritizes risk	20%
Secure remediation	Produces a sound patch and explains why it reduces risk without breaking functionality	20%
Validation and reflection	Verifies outcomes empirically and discusses uncertainty, ethics, and scope limits	20%

This rubric intentionally rewards reasoning quality and validation, not just "getting the right answer." That is important because LLM-era assessments must distinguish genuine understanding from polished but unverified output.

4. CONCLUSION

Recommended baseline governance is to align secure development with NIST's SSDF and AI use with its AI RMF; to use training targets and priority weakness categories from OWASP; and to structure threat and weakness language around taxonomies such as ATT&CK and CWE. In practice, that means every lab should specify the authorized target, expected evidence sources, allowed AI uses, and required validation steps before students begin [8-10].

For tooling, a defensible classroom baseline is National Security Agency's Ghidra for binary analysis; debuggers and syscall tracing for dynamic evidence; static analyzers and unit tests for secure remediation; and an institutionally governed LLM interface. If a copilot-style assistant is used, it should come with explicit responsible-use guidance, privacy controls, and a requirement that students submit prompt logs or AI-use disclosures [10-12].

For academic integrity, the key move is to assess process artifacts rather than only finished answers. Require individualized or randomized targets, evidence notebooks, short oral defenses, and a brief "what the model got right/wrong" reflection. That recommendation is supported by work showing both the educational upside of guided LLM use and the risk that generic chat access can lower engagement or encourage shallow task completion. Randomized reverse-engineering challenges are especially useful because they reduce answer sharing while preserving common learning objectives [13].

For safety and ethics, labs should run only in isolated VMs or containers with snapshot rollback, no live malware, no public targets, no institutional secrets in prompts, and no offensive deployment outside the classroom sandbox. In the United States, good-faith security-research exemptions are limited to lawful contexts and are not safe harbors against other laws. In the EU, observation, study, testing, and some decompilation are permitted only within specific legal conditions, especially interoperability and lawful use [14-19]. The course therefore needs an explicit rule: students may analyze only software and systems they own or are authorized to use for the lab's stated educational purpose.

The practical implication is straightforward: the most credible AI+RE curriculum is not the one with the most automation, but the one with the strongest verification culture [20-22]. Students should leave the course able to explain what they observed, what the model suggested, what they independently verified, what remained uncertain, and why their actions stayed within ethical and legal scope.

References

[1] Mekić, E. S., Jovanović, M. N., Kuk, K. V., Prlinčević, B. P., & Savić, A. M. (2024). Enhancing educational efficiency: Generative AI chatbots and DevOps in Education 4.0. *Computer Applications in Engineering Education*, *32*(6), e22804.

[2] Purkovic, S., Fetahovic, I., Mekic, E., Katipoglu, G., & Utku, S. (2024). Innovative Approach to Teaching Distributed Systems inEducation 4.0. *INTERNATIONAL JOURNAL OF ENGINEERING EDUCATION*, *40*(5).

[3] Muslić, E., Mekić, E., Novalić, F., Simović, V., & Katipoğlu, G. Educational–Industry Knowledge Graphs for Curriculum and Industry Alignment: A Case Study in Artificial Intelligence–Driven Digital Image Processing. *Available at SSRN 6619782*.

[4] Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., … OpenAI. (2023). *GPT-4 technical report*. *arXiv*. https://arxiv.org/abs/2303.08774

[5] Barone IV, C. R., Serafin, R., Ali-Gombe, A., Richard III, G. G., & Case, A. (2022). *A reverse engineering education needs analysis survey*. *arXiv*. https://arxiv.org/abs/2212.07531

[6] Biggs, J. (1996). Enhancing teaching through constructive alignment. *Higher Education, 32*(3), 347–364. https://doi.org/10.1007/BF00138871

[7] Chikofsky, E. J., & Cross II, J. H. (1990). Reverse engineering and design recovery: A taxonomy. *IEEE Software, 7*(1), 13–17. https://doi.org/10.1109/52.43044

[8] Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs. (2009). *Official Journal of the European Union*. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32009L0024

[9] GitHub Docs. (2026). *Responsible use of GitHub Copilot features*. https://docs.github.com/en/copilot/responsible-use-of-github-copilot-features

[10] He, J., & Vechev, M. (2023). Large language models for code: Security hardening and adversarial testing. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. https://arxiv.org/abs/2302.05319

[11] He, J., Vero, M., Krasnopolska, G., & Vechev, M.

(2024). *Instruction tuning for secure code generation*. *arXiv*. https://arxiv.org/abs/2402.09497

[12] Kagermann, H., Wahlster, W., & Helbig, J. (2013). *Recommendations for implementing the strategic initiative INDUSTRIE 4.0: Final report of the Industrie 4.0 Working Group*. acatech. https://en.acatech.de/publication/recommendations-for-implementing-the-strategic-initiative-industrie-4-0-final-report-of-the-industrie-4-0-working-group/

[13] Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., & Hoffmann, M. (2014). Industry 4.0. *Business & Information Systems Engineering, 6*(4), 239–242. https://doi.org/10.1007/s12599-014-0334-4

[14] Lyu, W., Wang, Y., Chung, T. R., Sun, Y., & Zhang, Y. (2024). Evaluating the effectiveness of LLMs in introductory computer science education: A semester-long field study. In *Proceedings of the ACM Conference on Learning @ Scale*. https://doi.org/10.1145/3657604.3662037

[15] National Institute of Standards and Technology (NIST). (2023). *Artificial intelligence risk management framework (AI RMF 1.0)* (NIST AI 100-1). https://doi.org/10.6028/NIST.AI.100-1

[16] National Institute of Standards and Technology (NIST). (2026). *NICE Framework Resource Center*. https://www.nist.gov/itl/applied-cybersecurity/nice/nice-framework-resource-center

[17] Nie, A., Chandak, Y., Suzara, M., Malik, A., Woodrow, J., Peng, M., Sahami, M., Brunskill, E., & Piech, C. (2025). The GPT surprise: Offering large language model chat in a massive coding class reduced engagement but increased adopters' exam performances. In *Proceedings of the ACM Conference on Learning @ Scale*. https://arxiv.org/abs/2407.09975

[18] OWASP Foundation. (2025). *OWASP Top 10:2025*. https://owasp.org/www-project-top-ten/

[19] Mekic, E., Djokic, I., Zejnelagic, S. and Matovic A. (2016), "Constructive approach in teaching of VoIP in line with good laboratory and manufacturing practice. Computer Applications in Engineering Education, ID CAE-15-117.R1, 24: 277–287. doi: 10.1002/cae.21705

[20] Souppaya, M., Scarfone, K., & Dodson, D. (2022). *Secure software development framework (SSDF) version 1.1: Recommendations for mitigating the risk of software vulnerabilities* (NIST SP 800-218). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-218

[21] Votipka, D., Rabin, S. M., Micinski, K., Foster, J. S., & Mazurek, M. L. (2019). An observational investigation of reverse engineers' processes. In *28th USENIX Security Symposium (USENIX Security 19)* (pp. 1875–1892). https://www.usenix.org/conference/usenixsecurity19/presentation/votipka

[22] Mekić, Edis, Irfan Fetahović, Kristijan Kuk, Brankica Popović, and Petar Čisar. "Data-driven educational decision-making model for curriculum optimization." In *Ethics in Online AI-based Systems*, pp. 97-118. Academic Press, 2024.

[23] Stojanović, M., Matović, A., Matović, M., Мекић, E., Baždar, V., &Maamar, Z. (2025). "Enhancing Al algorithm in Digital Signal Processing Education through Extreme Programming: Constructive Approach in IoP context". *International Journal of Engineering Education*,Vol.41,No.6,pp.1-15,2025, https://www.ijee.ie/contents/c410625.html 1.