**«Confirmed»**

**by the Head of the Department**

**of «Cryptology»**

**\_\_\_\_\_\_\_\_\_\_\_\_ Khudoykulov Z.T.**

**«\_\_\_\_»\_\_\_\_\_\_\_\_2024 y.**

**Questions of the final control work on the discipline**

**“Software security”**

1. Explain the basic concepts of information security (confidentiality, integrity, availability, identification, authentication, and authorization)
2. Explain information security issues
3. Explain the requirements for creating secure software
4. Explain security issues in software products
5. Software Development Life Cycle Concept: Communication
6. Software Development Life Cycle Concept: Requirement gathering
7. Software Development Life Cycle Concept: Feasibility study
8. Software Development Life Cycle Concept: System analysis
9. Software Development Life Cycle Concept: Software design
10. Software Development Life Cycle Concept: Coding
11. Software Development Life Cycle Concept: Testing
12. Software Development Life Cycle Concept: Integration
13. Software Development Life Cycle Concept: Implementation, deployment
14. Software Development Life Cycle Concept: Maintenance
15. Software Development Life Cycle Concept: Disposition
16. Software development Paradign
17. Explain the Waterfall model of software development
18. Explain the Iterative model of software development
19. Explain the V model of software development
20. What is the difference between the Waterfall and V models of software development
21. What is requirements engineering and what is done in the technical justification stage of requirements engineering
22. What is requirements engineering and what is done in the requirements gathering stage of requirements engineering
23. What is requirements engineering and what is done in the requirements specification stage of requirements engineering
24. What is requirements engineering and what is done in the requirements verification stage of requirements engineering
25. Functional and non-functional requirements of software tools
26. Security requirements imposed on software tools
27. What are the specific security requirements
28. SQUARE (Security Quality Requirements Engineering )
29. What is software design and what should it include
30. What is software design and what set of principles does it use
31. What is software design and what factors should be considered in it
32. Explain the levels of software design
33. Data flow diagrams and levels
34. Explain the components of DFD
35. Explain the components of DFD and level 0 DFD
36. Explain the components of DFD and level 1 DFD
37. Explain the difference between DFD levels 0 and 1
38. Describe the important design principles of software architecture
39. Software architecture styles: pipe and filter architecture
40. Software architecture styles: object-oriented architecture
41. Software architecture styles: client-server architecture
42. Software engine architecture styles: object broker architecture
43. Software architecture styles: event-driven architecture (implicit invocation)
44. Software tool architecture styles: layered architecture
45. Software tool architecture styles: storage architecture
46. Software tool architecture styles: process control architecture
47. What is a software vulnerability and why do they exist ?
48. What is a software vulnerability and why do they exist ? Explain the complexity of the vulnerability
49. What is a software vulnerability and why do they exist ? Explain the introduction of the vulnerability
50. What is a software vulnerability and why do they exist ? Explain the connection
51. What is a software vulnerability and why do they exist ? Explain the weaknesses in password management
52. Software vulnerabilities and their causes
53. Software Vulnerability: Buffer Overflow Explain with Examples
54. Software Vulnerability: Dandling Indicator Explain with Examples
55. Software Vulnerability: SQL Injection Explain with Examples
56. Software Vulnerability: OS Command Injection
57. Software Vulnerability: Integer Overflow Explain with examples
58. Software Vulnerability: Time-of-check-to-time-of-use (TOCTOU)
59. Vulnerabilities and protection measures announced by OWASP
60. What is the classification of threats and explain the accidental threat
61. Explain external and internal threats and the threat model
62. Explain the difference between active and passive threats
63. The threat model and its main purpose
64. STRIDE methodology
65. Security based on programming languages ​​and its capabilities
66. Programming Language-Based Security: Memory Security
67. Security based on programming languages: Type safety
68. Explain safe programming in C/C++ with examples (strcpy(), strcat(), snprintf())
69. Safe and unsafe programming languages
70. What is isolation (isolation, sandboxing) and explain it with examples
71. Controlling access in classic operating systems
72. General understanding of stack walking and its modifiers
73. Stack walking modifiers: Enable\_permission() explain with examples
74. Stack walking modifiers: Disable\_permission() explain with examples
75. Stack walking modifiers: Explain the difference between Enable\_permission() and Disable\_permission() with examples
76. Information leakage, confidentiality & integrity
77. Principles of information flow (NATO and Benelux classification)
78. What do you understand by clear information flow?
79. What do you understand by secret information flow?
80. Advantages and disadvantages of low-level programming languages
81. Advantages and disadvantages of high-level programming languages
82. What are the requirements for writing program code?
83. Recommendations for coding (naming conventions, formatting requirements, code structure, commenting procedures, error handling and response)
84. Pseudocode analysis (determining results by analyzing various given codes: for )

*begin  
numeric nNum1,nNum2, i, nSum  
display "ENTER THE FIRST NUMBER : "  
accept nNum1 // nNum1>5  
display "ENTER THE SECOND NUMBER : "  
accept nNum2 // nNum2>5*

*nSum=0*

*FOR i=2 TO nNum2 – 1*

*IF nNum2 MOD i ==0*

*nSum=nSum+1*

*END IF*

*END FOR*

*IF nSum==0*

*display " nNum1 " ELSE display " nNum2 "*

*END IF*

*END*

What function does the above pseudocode perform? Explain your answer.

1. Pseudocode analysis (determining results by analyzing various given codes: while)

*begin  
numeric nNum1,nNum2, i, nSum, j  
display "ENTER THE FIRST NUMBER : "  
accept nNum1 // nNum1> 5  
display "ENTER THE SECOND NUMBER : "  
accept nNum2 // nNum2>5*

*i=4*

*WHILE i< nNum2*

*nSum=0 j=2*

*WHILE j< i*

*IF i MOD j ==0*

*nSum=nSum+1*

*END IF*

*j=j+1*

*END WHILE*

*IF nSum==0 display i, " "*

*END IF*

*i=i+1*

*END WHILE*

*END*

What function does the above pseudocode perform? Explain your answer.

1. Pseudocode analysis (determining results by analyzing various given codes: array)

*begin  
numeric nNum1, i, max*

*numeric a[20]  
display "ENTER THE FIRST NUMBER : "  
accept nNum1 // nNum1> 5  
 FOR i=0 TO nNum1 – 1*

*accept a[i]*

*END FOR*

*max=a[1]*

*FOR i=0 TO nNum1 – 1*

*IF a[i] >max*

*max=max Else max=a[i]*

*END IF*

*END FOR*

*Output max END*

What function does the above pseudocode perform? Explain your answer.

1. Pseudocode analysis (determining results by analyzing various given codes: for)

*begin  
numeric nNum1,nNum2, i, nSum  
display "ENTER THE FIRST NUMBER : "  
accept nNum1 // nNum1>5  
display "ENTER THE SECOND NUMBER : "  
accept nNum2 // nNum2>5*

*nSum=0*

*FOR i=2 TO nNum2 – 1*

*IF nNum2 MOD i ==0 OR nNum1 MOD i ==0*

*nSum=nSum+1*

*END IF*

*END FOR*

*IF nSum==0*

*display " nNum1 " ELSE display " nNum2 "*

*END IF*

*END*

What function does the above pseudocode perform? Explain your answer.

1. Pseudocode analysis (determining results by analyzing various given codes: for)

*begin  
numeric nNum1,nNum2, i, nSum  
display "ENTER THE FIRST NUMBER : "  
accept nNum1 // nNum1>5  
display "ENTER THE SECOND NUMBER : "  
accept nNum2 // nNum2>5*

*nSum=0*

*FOR i=2 TO nNum2 – 1*

*IF nNum2 MOD i ==0 AND nNum1 MOD i ==0*

*nSum=nSum+1*

*END IF*

*END FOR*

*IF nSum==0*

*display " nNum1 " ELSE display " nNum2 "*

*END IF*

*END*

What function does the above pseudocode perform? Explain your answer.

1. Pseudocode analysis (determining results by analyzing various given codes: array)

*begin  
numeric nNum1, i,j,nSum*

*numeric a[5]  
display "ENTER THE FIRST NUMBER : "  
accept nNum1 // nNum1> 5*

*FOR i=0 TO nNum1 – 1*

*accept a[i]*

*END FOR*

*FOR i= 3 TO nNum1 – 1*

*FOR j= 3 TO nNum1 – 1*

*IF a[i] >a[j]*

*nSum=a[j] a[j]=a[i] a[i]=nSum*

*END IF*

*END FOR*

*END FOR*

*FOR i= 0 TO nNum1 – 1*

*Output a[i], " "*

*END FOR*

*END*

What function does the above pseudocode perform? Explain your answer.

1. Pseudocode analysis (determining results by analyzing various given codes: array)

*begin  
numeric nNum1, nNum2, i, min*

*numeric a[20]  
display "ENTER THE FIRST NUMBER : "  
accept nNum1 // nNum1> 5*

*display "ENTER THE SECOND NUMBER : "*

*accept nNum2 // nNum2> 5*

*FOR i=0 TO nNum1 – 1*

*accept a[i]*

*END FOR*

*min=a[0]*

*FOR i= 0 TO nNum1 – 1*

*IF a[i] >min*

*min=a[i] Else min=min*

*END IF*

*END FOR*

*Output min*

*END*What function does the above pseudocode perform? Explain your answer.

1. Pseudocode analysis (determining results by analyzing various given codes: internal cycle)

*begin  
numeric nNum1,nNum2, nSum, i, j  
display "ENTER THE FIRST NUMBER : "  
accept nNum1 // nNum1> 5  
display "ENTER THE SECOND NUMBER : "  
accept nNum2 // nNum2> 5  
FOR i=5 TO nNum2 – 1*

*nSum=0*

*FOR j=1 TO i – 1*

*IF i MOD j ==0*

*nSum=nSum+1*

*END IF*

*END FOR*

*IF nSum==1*

*display i, " "*

*END IF*

*END FOR*

*END*

What function does the above pseudocode perform? Explain your answer.

1. What is static analysis of a software tool?
2. How does static analysis of a software tool work?
3. What is static and dynamic analysis of a software tool?
4. MISRA Coding Guide
5. ISO 26262: ASIL standard
6. Software security testing, its types
7. Testing methods: explain black box testing
8. Testing methods: explain white box testing
9. Testing methods: explain gray box testing

**Written by:**

**the Senior Lecturer of  
the department of “Cryptology”: I.S. Olimov**