ОЛИМПИАДА «Я – МАГИСТР» ДЛЯ ПОСТУПАЮЩИХ В МАГИСТРАТУРУ

10.04.01 «Информационная безопасность» (программа «Искусственный интеллект в технологиях защиты информации»)

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ДЛЯ ПОДГОТОВКИ

К ЗАКЛЮЧИТЕЛЬНОМУ ЭТАПУ ОЛИМПИАДЫ

2025/2026 УЧЕБНОГО ГОДА

Составители: Чуйкова Е.Н., Айдинян А.Р., Ревякина Е.А.

(члены методической комиссии)

Председатель методической комиссии:

Чуйкова Е.Н.

ЗАКЛЮЧИТЕЛЬНЫЙ ЭТАП

Характер и уровень сложности олимпиадных задач направлены на достижение целей проведения Олимпиады: выявление и поддержка лиц, проявивших выдающиеся способности; стимулирование учебно-познавательной и научно-исследовательской деятельности обучающихся; развитие у обучающихся интеллектуальных и творческих способностей; создание необходимых условий для формирования качественного контингента магистрантов, ориентированных на продолжение академической карьеры; формирование системы непрерывного взаимодействия с одаренной и талантливой молодежью; распространение и популяризация научных знаний; привлечение талантливой молодежи, в том числе из зарубежных стран, к обучению в магистратуре.

Задания дифференцированы по сложности и требуют различных временных затрат на верное и полное решение. Задания направлены на выявление интеллектуального потенциала, аналитических способностей и креативности мышления участников и т.п.

Очный этап Олимпиады проводится только в письменной форме. Каждый участник Олимпиады получает бланк с заданием, содержащий 5 заданий. При выполнении заданий не допускается использование микропроцессорной техники. Можно иметь при себе 2 ручки синего или черного цвета (шариковая/ гелиевая/ перьевая), а также использовать карандаш и ластик.

При подготовке к Олимпиаде следует повторить приведенные ниже темы.

ПЕРЕЧЕНЬ ЭЛЕМЕНТОВ СОДЕРЖАНИЯ, ВКЛЮЧЕННЫХ В ЗАДАНИЯ ОЛИМПИАДЫ ЗАКЛЮЧИТЕЛЬНОГО ЭТАПА 2025/2026 УЧЕБНОГО ГОДА

Тема 1. Программирование

Содержит задания по программированию на языке C#.

Пример задания 1. Опишите, что делает следующая программа, и какой результат она выводит на экран:

```
static void function(int[] a, int l, int r)
{
    int x = a[(l + r) / 2];
    int i = l;
    int j = r;
    while (i <= j)
    {
        while (a[i] > x) i++;
        while (a[j] < x) j--;
        if (i <= j)
        {
            int temp = a[i];
            a[i] = a[j];
            a[j] = temp;
```

```
            i++;
            j--;
        }
    }
    if (i < r)
        function(a, i, r);

    if (l < j)
        function(a, l, j);
}

static void Main(string[] args)
{
    int[] m = {2,3,1};
    function(m, 0, m.Length-1);
    foreach (int e in m)
    {
        Console.Write(e + " ");
    }
}
```

Разбор задания

Эта программа реализует алгоритм быстрой сортировки (quicksort) для сортировки массива в порядке убывания.
Разберем по шагам:
1. В функции function принимается массив a и два индекса l и r, обозначающие левую и правую границы сортируемого участка.
2. Выбирается опорный элемент x как элемент, находящийся в середине участка (индекс (l + r) / 2).
3. Индексы i и j инициализируются как l и r соответственно.
4. Внешний цикл while (i <= j) выполняется до тех пор, пока i не станет больше j.
5. Внутри внешнего цикла:
   - Первый внутренний цикл while (a[i] > x) увеличивает i, пока элементы, на которые указывает i, больше опорного элемента x.
   - Второй внутренний цикл while (a[j] < x) уменьшает j, пока элементы, на которые указывает j, меньше опорного элемента x.
   - После этих циклов, если i <= j, то элементы a[i] и a[j] меняются местами, и индексы i и j сдвигаются навстречу друг другу (i++, j--).

Важно отметить: поскольку мы сортируем по убыванию, то в первом внутреннем цикле мы ищем элемент, который не больше опорного (останавливаемся, когда a[i] <= x), а во втором - элемент, который не меньше опорного (останавливаемся, когда a[j] >= x). Затем мы меняем их местами, чтобы большие элементы оказались слева, а меньшие справа.
6. После того, как внешний цикл завершится (когда i > j), массив окажется разделенным на две части:
   - левая часть (от l до j) содержит элементы, которые больше или равны опорному элементу (но не обязательно отсортированы),

- правая часть (от i до r) содержит элементы, которые меньше или равны опорному элементу.

7. Затем, если i < r, то рекурсивно вызывается function(a, i, r) для сортировки правой части.
8. Если l < j, то рекурсивно вызывается function(a, l, j) для сортировки левой части.
9. В функции Main создается массив m = {2,3,1} и вызывается function(m, 0, 2).
10. После сортировки массив выводится на консоль.

Поскольку алгоритм сортирует по убыванию, то массив {2,3,1} после сортировки станет {3,2,1}.
Программа выведет: 3 2 1
Как работает программа:

1. Функция function - это рекурсивная реализация QuickSort:

- **Выбор опорного элемента**: x = a[(l + r) / 2] (середина массива)
- **Разделение массива**:
  - Элементы **большие** опорного перемещаются в левую часть
  - Элементы **меньшие** опорного перемещаются в правую часть
- **Рекурсивная сортировка** двух подмассивов

2. Конкретные шаги для массива {2, 3, 1}:

Начальное состояние: [2, 3, 1]

1. **Первый вызов** function(m, 0, 2):
   - Опорный элемент: x = a[1] = 3
   - После разделения: [3, 2, 1]
   - Рекурсивные вызовы для подмассивов
2. **Рекурсивная обработка**:
   - Левый подмассив сортируется
   - Правый подмассив сортируется

Результат: [3, 2, 1]

Ответ: Программа представляет собой стандартный алгоритм сортировки половинным делением в порядке убывания.

Программа выводит на экран: 3 2 1.

Пример задания 2.

Дан исходный код программы. Опишите, что делает данная программа.

```
public partial class Form1 : Form

{

    Random r = new Random();

    CaptureDeviceList deviceList = CaptureDeviceList.Instance;

    ICaptureDevice captureDevice;
```

```
    public Form1()
    {
        InitializeComponent();
        captureDevice = deviceList[0];
        captureDevice.Open(DeviceMode.Promiscuous, 1000);


    }
    private void button1_Click(object sender, EventArgs e)
    {
        for (int i = 0; i < 8192; i++)
        {
            byte[] mac = new byte[6];
            for(int j=0; j<6; j++)
            {
                r.NextBytes(mac);
            }
            EthernetPacket ethPacket = new EthernetPacket(new
System.Net.NetworkInformation.PhysicalAddress(mac), new
System.Net.NetworkInformation.PhysicalAddress(new byte[6] { 123, 123, 123, 123, 123,
123 }), EthernetPacketType.IpV4);
            captureDevice.SendPacket(ethPacket);
        }
    }
}
```

Разбор задания.

Данная программа представляет собой форму приложения Windows Forms, которая отправляет Ethernet-пакеты через сетевое устройство.

Пошаговое объяснение:

1. **Инициализация формы и сетевого устройства:**

- Random r = new Random(); — создается генератор случайных чисел.
- CaptureDeviceList deviceList = CaptureDeviceList.Instance; — получается список сетевых устройств, доступных для захвата (и отправки) пакетов. Это часть библиотеки SharpPcap, которая используется для работы с сетевыми пакетами.
- ICaptureDevice captureDevice; — объявляется переменная для сетевого устройства.

В конструкторе формы Form1():

- InitializeComponent(); — инициализируются компоненты формы (кнопка и др.).

- captureDevice = deviceList[0]; — выбирается первое сетевое устройство из списка (обычно это первая сетевая карта).
- captureDevice.Open(DeviceMode.Promiscuous, 1000); — устройство открывается в промис-куозном режиме с таймаутом 1000 мс. Промис-куозный режим позволяет устройству захватывать все пакеты в сети, а не только адресованные ему.

2. **Обработчик нажатия кнопки button1_Click:**

- При нажатии кнопки выполняется цикл 8192 раза (от 0 до 8191).
- В каждой итерации создается новый Ethernet-пакет и отправляется через выбранное сетевое устройство.

Процесс создания и отправки пакета:

- byte[] mac = new byte[6]; — создается массив из 6 байт для MAC-адреса.
- r.NextBytes(mac); — заполняет массив mac случайными байтами.
- EthernetPacket ethPacket = new EthernetPacket(...); — создается Ethernet-пакет.
  - Источник (отправитель) — случайный MAC-адрес из массива mac.
  - Назначение (получатель) — MAC-адрес { 123, 123, 123, 123, 123, 123 } (в шестнадцатеричном формате это 7B:7B:7B:7B:7B:7B).
  - Тип пакета — EthernetPacketType.IpV4 (указывает, что внутри Ethernet-пакета находится IPv4-пакет, но в данном случае содержимое не заполняется, так как пакет создается только на Ethernet-уровне).
- captureDevice.SendPacket(ethPacket); — отправляет созданный Ethernet-пакет через сетевое устройство.

Что делает программа в целом:
При каждом нажатии кнопки программа отправляет 8192 Ethernet-пакета с случайным MAC-адресом отправителя и фиксированным MAC-адресом получателя (123.123.123.123.123.123). Тип пакета указан как IPv4, но сам IP-пакет (или любой другой полезный груз) не создается. Таким образом, это просто Ethernet-фреймы без полезной нагрузки.

Эта программа создает сетевой трафик, который можно классифицировать как **сетевую атаку типа "флуд MAC-адресами"**.

Подробный анализ:

1. **Инициализация:**

- Создается генератор случайных чисел
- Получается список сетевых устройств (сетевых карт)
- В конструкторе форма открывается первое сетевое устройство в промис-куозном режиме для захвата/отправки пакетов

2. **При нажатии кнопки (button1_Click):**

- **8192 раза** отправляются Ethernet-пакеты
- Для каждого пакета:

- Генерируется случайный MAC-адрес отправителя (6 случайных байт)
- MAC-адрес получателя фиксированный: {123, 123, 123, 123, 123, 123}
- Тип пакета указывается как IPv4
- Пакет отправляется через сетевой интерфейс

Цель и последствия:

Это атака на коммутаторы (свитчи):

- Коммутаторы хранят таблицу MAC-адресов (CAM-table), сопоставляя MAC-адреса с портами
- При заполнении этой таблиицы фейковыми MAC-адресами (8192 случайных адреса), коммутатор может:
  - Переполнить таблицу и начать работать как хаб (транслировать трафик на все порты)
  - Зависнуть или перезагрузиться
  - Отказать в обслуживании

Это вредоносный код, который:

1. Создает искусственный сетевой трафик
2. Может нарушить работу сетевого оборудования
3. Может использоваться для прослушивания трафика (если коммутатор перейдет в режим хаба)
4. Является формой DoS-атаки (Denial of Service)

Ответ: осуществляет атаку на коммутатор путём переполнения таблицы сопоставления MAC-адресов и интерфейсов: генерирует 8192 пакета со случайными MAC-адресами отправителя.

Тема 2. Запросы на языке SQL

Содержит задания по формированию запросов к базе данных на языке SQL.

Пример задания 1.

В базе данных есть следующие таблицы:

1. Студенты (students):

```
CREATE TABLE students (
   student_id INT PRIMARY KEY,
   first_name VARCHAR(50),
   last_name VARCHAR(50),
   group_code VARCHAR(10));
```

2. Экзамены (exams):

```
CREATE TABLE exams (
   exam_id INT PRIMARY KEY,
   student_id INT,
   exam_date DATE,
   score INT CHECK (score BETWEEN 0 AND 100),
   FOREIGN KEY (student_id) REFERENCES students(student_id));
```

Данные в таблицах:

students

student_id	first_name	last_name	group_code
1	Иван	Петров	IT-101
2	Мария	Сидорова	IT-101
3	Алексей	Иванов	IT-102
4	Екатерина	Смирнова	IT-102
5	Дмитрий	Кузнецов	IT-101

exams

exam_id	student_id	exam_date	score
1	1	2024-01-10	85
2	1	2024-01-20	90
3	2	2024-01-10	78
4	2	2024-01-20	92
5	3	2024-01-10	45
6	3	2024-01-20	60
7	4	2024-01-10	95
8	4	2024-01-20	88
9	5	2024-01-10	30
10	5	2024-01-20	NULL

Примечание: NULL у студента 5 означает, что он не явился на пересдачу.

Сформулируйте на языке SQL запрос, который выведет фамилию, имя и средний балл для каждого студента (студентов без оценок (все оценки NULL) не учитывать) и приведите результат выполнения запроса.

Ответ:

Запрос на языке SQL:

```
SELECT students.last_name,    students.first_name,    AVG(exams.score) as avg_
score
FROM students JOIN exams ON students.student_id = exams.student_id
WHERE exams.score IS NOT NULL
GROUP BY students.student_id, students.last_name, students.first_name
ORDER BY avg_score DESC;
```

Результат запроса:

last_name	first_name	avg_score
Смирнова	Екатерина	91.50
Петров	Иван	87.50
Сидорова	Мария	85.00
Иванов	Алексей	52.50
Кузнецов	Дмитрий	30.00

Пример задания 2.

Опишите, что вычисляет следующий запрос на языке SQL и приведите результат запроса:

```
SELECT    students.group_code,  exams.score as group_avg_score
FROM students JOIN exams ON students.student_id = exams.student_id
WHERE exams.score IS NOT NULL
GROUP BY students.group_code
HAVING AVG(e.score) > 70
ORDER BY group_avg_score DESC;
```

Ответ:

Запрос выводит код группы и средний балл по группе, но только для тех групп, где средний балл больше 70 (студенты без оценок (все оценки NULL) не учитываются). Результат запроса отсортирован по убыванию среднего балла.

Результат запроса:

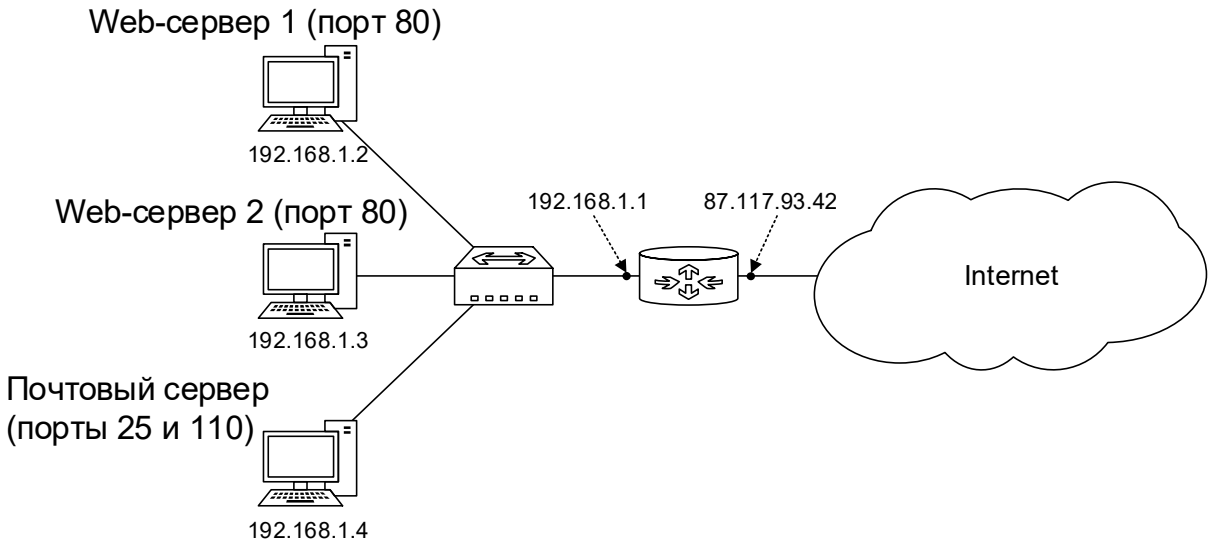
group_code	group_avg_score
IT-102	77.00
IT-101	73.25

Тема 3. Компьютерные сети.

Включает задания по планированию IP-сетей, настройке портов (протокол NAT) и маршрутизации в компьютерной сети.

Пример задания 1.

Для схемы, приведённой на рисунке, укажите параметры настройки перенаправления портов для того, чтобы обеспечить доступность из внешней сети для двух web-серверов и одного почтового сервера во внутренней сети.



Приведите конфигурацию маршрутизатора в виде таблицы следующего формата:

Внутренний IP-адрес	Внутренний порт	Внешний порт	Внешний IP-адрес	Сервис
…	…	…	…	…

Ответ:

Внутренний IP-адрес	Внутренний порт	Внешний порт	Внешний IP-адрес	Сервис
192.168.1.2	80	80	87.117.93.42	Web-сервер 1
192.168.1.3	80	Любой, кроме 80, 25 и 110	87.117.93.42	Web-сервер 2
192.168.1.4	25	25	87.117.93.42	Почтовый сервер
192.168.1.4	110	110	87.117.93.42	

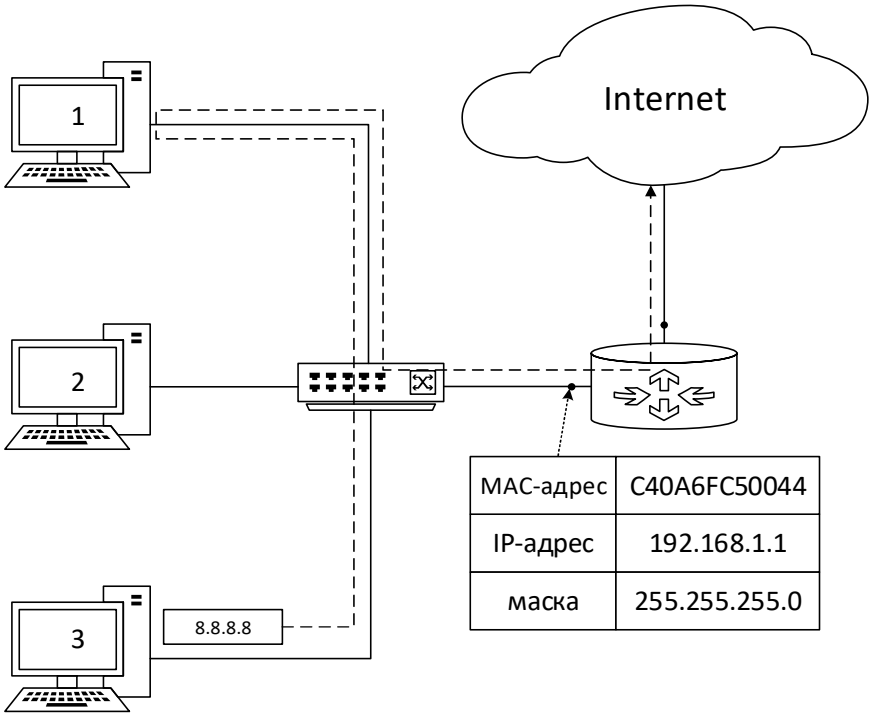
Пример задания 2.

Для схемы, приведённой на рисунке, заполните пустые поля в таблицах с настройками, чтобы пакет, отправленный с компьютера № 3 на адрес 8.8.8.8 прошёл по пути, показанному пунктирной стрелкой. Укажите IP и MAC-адрес источника, записанные в заголовке этого пакета при его приходе на маршрутизатор.

MAC-адрес	D805E65239B1
IP-адрес	
маска	
шлюз	

MAC-адрес	E60A56F73B12
IP-адрес	192.168.1.3
маска	255.255.255.0
шлюз	

MAC-адрес	B75A409FF32C
IP-адрес	
маска	
шлюз	

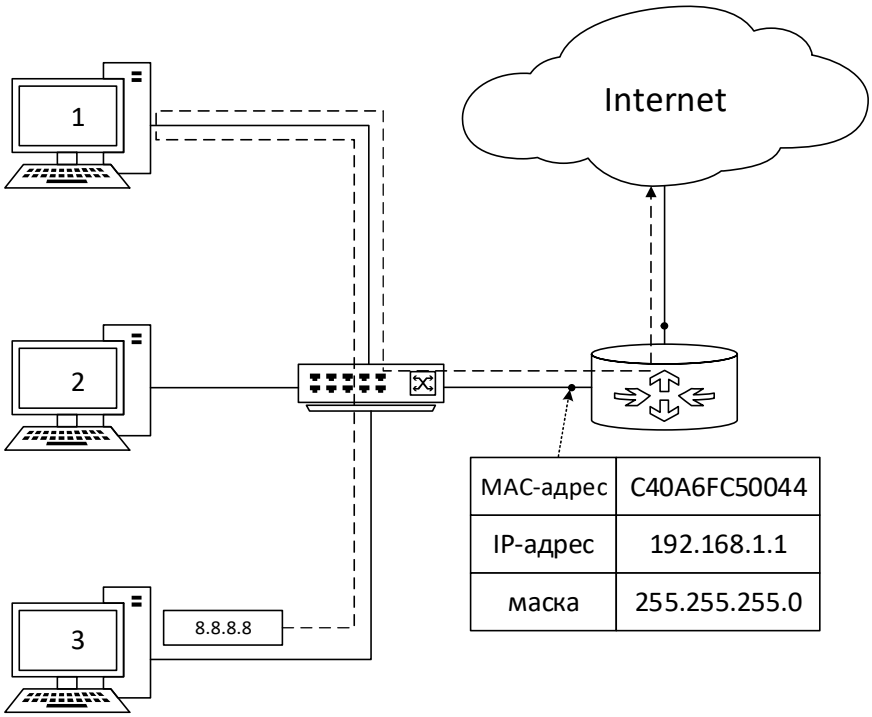


Ответ:

MAC-адрес	D805E65239B1
IP-адрес	192.168.1.2
маска	255.255.255.0
шлюз	192.168.1.1

MAC-адрес	E60A56F73B12
IP-адрес	192.168.1.3
маска	255.255.255.0
шлюз	без разницы

MAC-адрес	B75A409FF32C
IP-адрес	192.168.1.4
маска	255.255.255.0
шлюз	192.168.1.2



IP и MAC-адрес источника, записанные в заголовке пакета, отправленного с компьютера № 3 на адрес 8.8.8.8, при его приходе на маршрутизатор: IP — 192.168.1.2, MAC — D805E65239B1.

Тема 4. Информационная безопасность.

Содержит задания по кодированию передаваемых сообщений, анализу подозрительной активности в компьютерной сети.

Пример задания 1.

Абонент А передал абоненту В сообщение, закодированное в соответствии с табл. 1. В результате была получена последовательность бит открытого текста $O(i)$, $i = 1,\ldots,33$.

Таблица 1

№	символ	код	№	символ	код	№	символ	код
1	***а***	010	12	***к***	100	23	***х***	110101
2	***б***	000	13	***л***	0111100	24	***ц***	110110
3	***в***	011000	14	***м***	0111101	25	***ч***	110111
4	***г***	011001	15	***н***	0111110	26	***ш***	111000
5	***д***	101	16	***о***	0111111	27	***щ***	111001
6	***е***	011010	17	***п***	110000	28	***ъ***	111010
7	***ё***	011011	18	***р***	001	29	***ы***	111011
8	***ж***	0111000	19	***с***	110001	30	***ь***	111100
9	***з***	0111001	20	***т***	110010	31	***э***	111101
10	***и***	0111010	21	***у***	110011	32	***ю***	111110
11	***й***	0111011	22	***ф***	110100	33	***я***	111111

Затем было произведено преобразование закодированного сообщения: $S(i) = [O(i) + S(i-1)] \bmod 2$, $S(0) = 0$ (см. рис. 1).

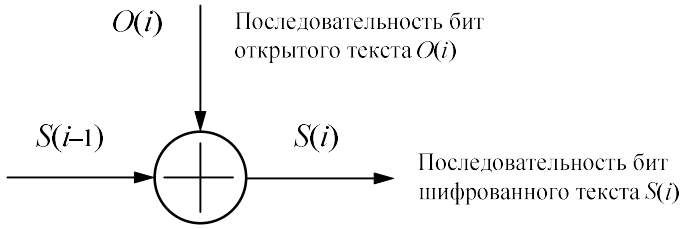


Рисунок 1 — Схема преобразования закодированного сообщения.

В результате была получена последовательность бит шифрованного текста $S(i)$, $i = 1,\ldots,33$:

001111111001100001100110000000 1100

Какое сообщение абонент А передал абоненту В?

Ответ:

Согласно схеме шифрования i-ый бит исходного открытого сообщения $O(i)$ равен сумме $(S(i) + S(i\text{-}1)) \bmod 2$ для $i>0$. Учитывая, что $S(0) = 0$ и $S(1) = O(1)$ получаем последовательность бит открытого сообщения i=1,…,33

01000000101010001010101000000101 0.

Воспользуемся таблицей кодов. Последовательность бит 010 соответствует символу ***а***. Последующие три бита 000 – символу ***б***. Рассуждая аналогично получаем:
010 – ***а***
000 – ***б***
001 – ***р***
010 – ***а***
100 – ***к***
010 – ***а***
101 – ***д***
010 – ***а***
000 – ***б***
001 – ***р***
010 – ***а***
Таким образом, было закодировано слово *абракадабра.*

Пример задания 2.

Вы — стажёр в SOC (Security Operations Center). Система обнаружения вторжений (IDS) выдала предупреждение о подозрительном сетевом трафике с IP-адреса 192.168.1.105. Вам передали фрагмент этого трафика в формате PCAP. Из-за политики безопасности сам файл передать нельзя, но специалист предоставил текстовый дамп нескольких пакетов, полученных с помощью утилиты tcpdump.

Фрагмент дампа трафика:

```
18:45:22.123456 IP 192.168.1.105.54321 > 10.0.5.20.22: Flags [S], seq 1000, win 29200
18:45:22.234567 IP 192.168.1.105.54322 > 10.0.5.20.22: Flags [S], seq 2000, win 29200
18:45:22.345678 IP 192.168.1.105.54323 > 10.0.5.20.22: Flags [S], seq 3000, win 29200
18:45:22.456789 IP 192.168.1.105.54324 > 10.0.5.20.22: Flags [S], seq 4000, win 29200
18:45:22.567890 IP 192.168.1.105.54325 > 10.0.5.20.22: Flags [S], seq 5000, win 29200
18:45:23.123456 IP 192.168.1.105.54321 > 10.0.5.20.22: Flags [S], seq 6000, win 29200
18:45:23.234567 IP 192.168.1.105.54326 > 10.0.5.20.22: Flags [S], seq 7000, win 29200
18:45:24.123456 IP 192.168.1.105.54327 > 10.0.5.20.22: Flags [S], seq 8000, win 29200
18:45:25.123456 IP 192.168.1.105.54328 > 10.0.5.20.22: Flags [S], seq 9000, win 29200
```

Дополнительная информация:

- Сервер 10.0.5.20 — корпоративный SSH-сервер.
- В нормальном режиме с 192.168.1.105 на SSH-сервер устанавливается 1-2 соединения в день.

Требуется выполнить следующее:

1. Анализ трафика: Что означают флаги [S] в пакетах? Какой протокол транспортного уровня используется?
2. Идентификация атаки: Какой тип сетевой атаки представлен в дампе? Объясните по каким признакам вы это определили.
3. Определение цели атаки: Какую цель преследует злоумышленник, осуществляя такую атаку?
4. Меры защиты: Предложите две технические меры защиты от подобных атак на сетевом уровне.

Разбор задания

1. Анализ трафика

- Флаги **[S]** — это флаг SYN в TCP-сегменте. Он указывает на попытку установления TCP-соединения (первый шаг трёхстороннего рукопожатия: SYN → SYN-ACK → ACK).
- Протокол транспортного уровня — TCP (об этом говорит наличие флагов и номеров последовательности seq).

2. Идентификация атаки

Тип атаки: SYN-флуд**,** разновидность DoS-атаки (отказ в обслуживании).

Признаки:

1. Массовые SYN-запросы: За короткий промежуток времени (4 секунды) с одного IP-адреса (192.168.1.105) отправлено 9 TCP-пакетов с флагом SYN.
2. Изменение портов источника: Для каждого нового соединения (кроме одного повторного) используется новый порт-источник (54321, 54322, 54323 и т.д.). Это попытка обойти простые механизмы блокировки.
3. Цель — порт 22 (SSH): Все запросы направлены на порт 22, который используется для SSH. Это критичная служба.
4. Отсутствие завершения рукопожатия: В дампе видны только SYN-пакеты. Нет пакетов с флагами [R] (RST) или [F] (FIN) для завершения соединений.

3. Цель атаки

Цель злоумышленника — исчерпать ресурсы сервера (таблицу полуоткрытых соединений). Сервер, получая SYN, выделяет память под новое соединение и отправляет SYN-ACK в ответ, ожидая завершающий ACK от клиента. Если

злоумышленник не отправляет финальный ACK (или использует фальшивый IP), соединение остаётся в состоянии SYN_RECEIVED до таймаута. При большом количестве таких запросов очередь переполняется, и сервер перестаёт принимать новые легитимные подключения к SSH, вызывая отказ в обслуживании.

4. Меры защиты

1. Включение механизма SYN Cookies на сервере. При подозрении на атаку сервер не выделяет память под новое соединение сразу, а кодирует информацию в специальном «печенье» (cookie), которое отправляет клиенту. Ресурсы выделяются только при получении валидного ответного ACK с этим cookie.
2. Настройка правил фильтрации на межсетевом экране (файрволе):

   - Ограничение числа одновременных SYN-соединений с одного IP-адреса.
   - Применение лимитов на скорость для SYN-пакетов.
   - Использование черных списков для IP-адресов, проявляющих подозрительную активность.

Ответ:

1. Флаги — флаг SYN протокола TCP, означающий запрос на установление соединения. Используется протокол TCP.
2. Тип атаки — SYN-флуд (SYN Flood). Признаки: множество TCP SYN-пакетов за короткое время с одного источника на один порт (22/SSH), при этом источник использует разные порты для обхода блокировок.
3. Цель атаки — вызвать отказ в обслуживании (DoS) SSH-сервера путём исчерпания его ресурсов (заполнения таблицы полуоткрытых соединений).
4. Меры защиты:

   - Включение SYN Cookies на сервере.
   - Настройка на сетевом оборудовании ограничений на частоту SYN-пакетов и количество соединений с одного IP.

Литература для подготовки

1. Залогова Л.А., Основы объектно-ориентированного программирования на базе языка C#. – М.: Лань, 2018. – 192 с.
2. Полищук Ю.В., Боровский А.С. Базы данных и их безопасность. - М: ООО "Научно-издательский центр ИНФРА-М", 2022. – 210 с.
3. Руденков Н.А. Технологии защиты информации в компьютерных сетях. – М.: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021. – 368 с.

Приложение 4

Заключительный этап олимпиады «Я – магистр» для поступающих в магистратуру в 2026 году

Олимпиада по 10.04.01 «Информационная безопасность» (программа «Искусственный интеллект в технологиях защиты информации»)

Критерии проверки.

Вариант заключительного этапа Олимпиады по 10.04.01 «Информационная безопасность» (программа «Искусственный интеллект в технологиях защиты информации») включает в себя 5 заданий разного типа. Каждое задание оценивается от 0 до 20 баллов. Наибольшая итоговая сумма баллов, которой могут быть оценены ответы на все вопросы олимпиадного варианта при условии отсутствия в них ошибок, неправильных, неполных или неточных ответов, равна 100. Неверные ответы оцениваются в 0 баллов. Возможен частичный зачёт баллов за неполный ответ на задание. Под неполным понимается ответ, содержащий правильные ответы не на все вопросы задания. В таком случае присуждается только часть баллов за правильные ответы задания, соответствующая доле от максимально возможного балла. Подсчёт итоговой оценки за задание осуществляется путём суммирования баллов, выставленных за каждый из вопросов.

Задача 1.

Всего баллов: 20.

Задан программный код на языке C#.

Определите, какую задачу решает данная программа, и опишите пошаговый алгоритм её работы.

Критерии проверки (оценивания)

Критерий	Баллы	Описание
1. Корректное определение задачи программы	7 баллов	Чётко, точно и полно указано, что программа выполняет. За каждый правильный пункт по **1 баллу**.
2. Правильное описание последовательности алгоритма	4 балла	За каждый пункт на правильном месте в последовательности — **0,5 балла**.
3. Правильное описание назначения регулярного выражения	2 балла	Правильно указано, для чего используется регулярное выражение. Ошибочное утверждение— **0 баллов**. Отсутствие термина «регулярное выражение» — **минус 0,5 балла**.
4. Точность описания структуры данных и удаления дубликатов	1 балл	Недопустима путаница со списками, массивами или словарями.
5. Корректное описание сортировки и использования LINQ	1 балл	Не упомянуто понятие «LINQ» — минус 0,5 балла.
6. Полнота описания вывода результата	1 балл	Верно описан формат вывода. Не указано, что вывод идёт в консоль — минус 0,5 балла.
7. Учёт обработки ошибок	1 балл	Правильно описан порядок обработки ошибок.
8. Пример и его корректность	1 балл	Приведён корректный пример входных данных и ожидаемого результата, соответствующий логике программы.
9. Отсутствие технических ошибок и неточностей	1 балл	В ответе отсутствуют ложные или противоречивые утверждения о работе программы.
10. Ясность, структурированность и логичность изложения	1 балл	Ответ логично структурирован, легко читается, шаги чётко выделены, отсутствует путаница и избыточная информация.

Задача 2.

Всего баллов: 20.

Указана структура и содержимое таблиц базы данных. Представлен запрос к базе данных на языке SQL.

Определите, что вычисляет данный запрос, какие сведения выводит и каким будет результат запроса.

Критерии проверки (оценивания)

Критерий	Баллы	Описание
Корректность определения смысла запроса	7	Правильно описано, что вычисляет запрос - 7 баллов. Неправильно описано или не описано, что вычисляет запрос - 0 баллов.
Правильность определения перечня сведений, выводимых запросом	3	Правильно перечислены все сведения, которые выводит запрос – 3 балла. Перечислены не все сведения, которые выводит запрос – 1 балл. Неверно указаны или не указаны выводимые запросом сведения – 0 баллов.
Правильность определения результата запроса	10	Результат запроса (в виде таблицы) приведен, но отсутствуют некоторые столбцы, имеются неверные значения столбцов – 3 балла. Результат запроса (в виде таблицы) приведен, но отсутствуют некоторые столбцы – 5 баллов. Результат запроса (в виде таблицы) приведен, но имеются неверные значения в столбцах – 7 баллов. Приведен правильный результат запроса (в виде таблицы) – 10 баллов.

Задача 3.

Всего баллов: 20.

Дана схема сети. На ней изображен пакет и путь его передачи. Заполните отсутствующую в заголовке пакета информацию.

Критерии проверки (оценивания)

Критерий	Баллы	Описание
Корректность заполнения предложенной таблицы	20	2,5 балла за каждую правильно заполненную ячейку таблицы

Задача 4.

Всего баллов: 20.

В сетевом трафике система обнаружения вторжений (СОВ) зафиксировала подозрительную последовательность пакетов с определенным содержимым. На основе анализа указанной последовательности пакетов определите вид атаки, цель атаки, механизм смягчения последствий атаки.

Критерии проверки (оценивания)

Критерий	Баллы	Описание
Корректность определения вида атаки	5	Правильно определено название атаки - 7 баллов. Неправильно определен вид атаки/нет определения вида атаки - 0 баллов.
Правильность определения цели атаки	5	Правильно определена цель атаки - 5 баллов. Неправильно определена цель атаки/нет определения цели атаки - 0 баллов.
Правильность определения механизма смягчения последствий атаки	10	Правильное и полное определение способов смягчения последствий атаки – 10 баллов. Правильное, но неполное определение способов смягчения последствий атаки – 5 баллов. Неправильное определение способов смягчения последствий атаки /нет определения способов смягчения последствий атаки - 0 баллов.

Задача 5.

Всего баллов: 20.

Абонент А передал абоненту В сообщение, закодированное в соответствии с заданной кодовой таблицей. В результате получена последовательность бит открытого текста Р. Затем закодированное сообщение по указанному правилу преобразовано в последовательность бит шифрованного текста С.

Определите, какое сообщение абонент А передал абоненту В.

Требуется:

1. Восстановить исходную битовую последовательность открытого текста Р из последовательности бит шифрованного текста С.
2. Декодировать последовательность Р в текст, используя кодовую таблицу.
3. Записать расшифрованное сообщение.

Критерии проверки (оценивания)

Критерий	Баллы	Описание
Корректность восстановления исходной битовой последовательности	7	Правильно определено обратное преобразование и корректно применено к шифртексту *C*, получена последовательность *P* – 7 баллов. Преобразование определено верно, но допущена арифметическая ошибка (не более 2 бит) – 4 балла. Неверное преобразование или грубые ошибки – 0 баллов.
Корректность декодирования битовой последовательности	7	Последовательность *P* корректно разбита на коды согласно таблице, все символы распознаны верно – 7 баллов. Допущены 1-2 ошибки в разбиении кодов – 3-4 балла. Разбиение выполнено частично, угадано не более 30% символов – 2 балла. Декодирование не выполнено или полностью неверное – 0 баллов.
Правильность итогового сообщения	6	Получено корректное сообщение – 6 баллов. Сообщение неверное или отсутствует – 0 баллов.