Федеральное государственное автономное образовательное учреждение высшего образования
Национальный исследовательский университет
«Высшая школа экономики»
Московский институт электроники и математики им. А.Н. Тихонова

Методические рекомендации для проведения теоретического этапа Московского конкурса межпредметных навыков и знаний «Интеллектуальный мегаполис. Потенциал» в номинации «ИТ-класс» для направлений «Создание цифровых двойников», «Большие данные и технологии искусственного интеллекта», «Робототехника», «Информационная безопасность и технологии связи» (профиль Информатика)

Москва, НИУ ВШЭ 2024-2025 г.

Теоретический этап Конкурса проводится в очном дистанционном формате с использованием технологии прокторинга. Участникам необходимо иметь компьютер (ПК или ноутбук; прохождение диагностики на мобильных устройствах - невозможно) с выходом в Интернет, веб-камерой и микрофоном, а также смартфон (или планшет) со стабильным интернетом и приложением для считывания QR-кодов. Требуется предварительная настройка оборудования: https://im.mcko.ru/docs/Инструкция_для_участника_конкурса_Интеллектуальный_мегаполис_Потенциал.pdf . Браузер разрешается использовать только для прохождения заданий этапа и процедуры прокторинга.
Дополнительное ПО, разрешенное для прохождения: текстовый редактор, графический редактор, MS Excel, электронные таблицы (как обычный калькулятор, исключая специализированные формулы), обычный встроенный калькулятор.
Чем пользоваться категорически нельзя (ведет к отклонению работы): веб-поиском, методическими указаниями.
На выполнение заданий *теоретического* этапа Конкурса отводится 120 минут. Во время проведения мероприятия участник может выйти из зоны проведения мероприятия не более чем на 5 минут, предупредив *проктора на камеру*. Мероприятие не продлевается на время отсутствия участника.

План конкурсных материалов для проведения *теоретического* этапа Конкурса

№ задания	**Уровень сложности**	***Уникальные кодификаторы Конкурса***	**Контролируемые требования к проверяемым умениям**	**Балл**
1.	Базовый	2.3 Сравнение чисел, записанных в двоичной, восьмеричной и шестнадцатеричной системах счисления и выполнение с ними арифметических действий	Умение переводить целые числа из двоичной системы счисления в восьмеричную и шестнадцатеричную и обратно. Умение выполнять арифметические операции с целыми числами в двоичной, восьмиричной и шестнадцатиричной системах счисления.	6
2.	Базовый	4.1 Операционная система. Файловая система. Операции с каталогами и файлами 4.2 Поиск в файловой системе	Умение выполнять операции с каталогами и файлами. Умение осуществлять поиск в файловой системе.	6
3	Базовый	3.1 Принципы построения компьютерных сетей. Сетевые протоколы. Адресация в сети Интернет	Умение применять принципы построения компьютерных сетей. Умение применять сетевые -протоколы. -Умение применять знания об адресации в сети Интернет	6
4	Повышенный	3.6 Процедуры и функции. Передача параметров. Локальные и глобальные объекты. Рекурсия 3.7 Одномерные массивы, их обработка, суммирование элементов, поиск элемента по условию. Обработка двумерных массивов 3.8	Умение применять процедуры и функции. Умение применять передачу параметров. Умение применять рекурсию. Имение работать с одномерными массивами. Умение выполнять операции со стеком. Умение выполнять операции с	10

		Стек. Операции со стеком. Стек и рекурсия. Вычисление значения выражения в польской инверсной записи. Задача о Ханойских башнях 3.9 Очередь. Операции с очередью 3.10 Линейный список. Операции с линейным списком	очередью. Умение выполнять операции с линейным списком.	
5	Повышенный	1.2 Элементы комбинаторики. Принцип включения и исключения	Умение применять принцип включения и исключения.	10

Демонстрационный вариант конкурсных заданий *теоретического* этапа Конкурса
Пример состава задания теоретического этапа Конкурса.

1. *Даны четыре числа 572_8, 1011101_2, $2F3_{16}$, $1A7_{16}$. Выберите наибольшее из них в десятичной системе счисления:*
   А) 993
   Б) 755
   В) 378
   Г) 823

Ответ. Б.

Теория:

В десятичной система счисления, где основанием является 10, используются цифры от 0 до 9.
В двоичной системе счисления, где основанием является 2, используются только две цифры, 0 и 1.
В шестнадцатеричной системе счисления, где основанием является 16, используются цифры от 0 до 9 и символы A, B, C, D, E, F, где A = 10, B = 11 и т.д.
Преобразование из других систем счисления в десятичную осуществляется с использованием формулы:

$$N_{10} = \sum_{i=0}^{n-1} d_i \cdot b^i$$

где d — цифра числа, b — основание системы счисления, i — индекс разряда.

Решение:

Даны числа: 572_8, 1011101_2, $2F3_{16}$, $1A7_{16}$. Переведем их все для удобства в десятичную систему счисления:

$$572_8 = 5 \cdot 8^2 + 7 \cdot 8^1 + 2 \cdot 8^0 = 378$$
$$1011101_2 = 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 93$$
$$2F3_{16} = 2 \cdot 16^2 + 15 \cdot 16^1 + 3 \cdot 16^0 = 755$$
$$1A7_{16} = 1 \cdot 16^2 + 10 \cdot 16^1 + 7 \cdot 16^0 = 423$$

Таким образом, наибольшее число 755.

Методика оценки заданий: если выбран верный ответ, участник получает 6 баллов.
Если выбран неверный ответ, участник получает 0 баллов.

Описание возможных трудностей при подготовке: во избежание трудностей необходимо ознакомиться с теоретической вставкой к задаче, так как важно знать правила перевода из n-й системы счисления в десятичную.

Разбор типичных ошибок: типичные ошибки могут возникнуть, если не повторить формулу для перевода из n-й системы счисления в десятичную, так же внимательно перепроверяйте алгебраические расчеты во избежание досадных ошибок из-за неверных вычислений.

2. На уроке информатики ученику необходимо в Unix-подобной системе создать внутри домашнего каталога с именем «projects» вложенный каталог «2024». Как должна выглядеть последовательность команд?
   А) mkdir ~/projects/2024
   Б) mkdir ~/projects && mkdir ~/2024
   В) mkdir -p ~/projects/2024
   Г) mkdir projects/2024

Ответ: А

Теория:
При решении подобной задачи важно помнить, что в Unix-подобных операционных системах каталоги и файлы организуются в виде иерархической файловой структуры. У пользователя есть возможность создавать, удалять и управлять этими каталогами при помощи команд в терминале.
Для облегчения работы с директориями, каталогами и файлами в Unix-подобных системах стоить запомнить несколько пунктов, которые помогут глубже разобраться в управлении файловой системой и автоматизации процессов:
1. `~` — символ домашней директории пользователя.
2. `mkdir` — команда для создания нового каталога ("make directory").
3. `-p` — опция для создания цепочки вложенных каталогов, если они не существуют. Без этой опции команда вернёт ошибку, если хотя бы один каталог в цепочке не существует.
4. `/` — символ разделителя каталогов в пути.
5. `cd` — команда для смены текущего каталога ("change directory").
6. `pwd` — команда для отображения полного пути к текущему каталогу ("print working directory").
7. `ls` — команда для отображения содержимого каталога.
8. `{}` — фигурные скобки используются для создания нескольких каталогов сразу. Пример: команда mkdir ~/projects/{frontend,backend,design} создаст каталоги "frontend", "backend" и "design" внутри "projects".
9. `rmdir` — команда для удаления пустого каталога.
10. `rm -r` — команда для удаления каталога и всего его содержимого (рекурсивно).
11. `cp -r` — команда для копирования каталога и его содержимого (рекурсивно).
12. `mv` — команда для перемещения или переименования файлов и каталогов.
13. `" "` — если имя каталога или файла содержит пробелы, его нужно взять в кавычки, чтобы команда понимала, что это одно имя, а не несколько аргументов.
14. `.` (точка) — текущий каталог, используется для ссылки на текущую директорию в командах.
15. `chmod` — команда для изменения прав доступа к файлам и каталогам.
16. `-a` — опция для команды ls, которая показывает все файлы, включая скрытые (файлы и каталоги, имена которых начинаются с точки).
В Unix-системах права доступа задаются с помощью трёх цифр, каждая из которых определяет разрешения для трёх групп пользователей: владельца (user), группы (group) и остальных (others). Каждая цифра является суммой разрешений, где 4 = чтение (r), 2 = запись (w), 1 = выполнение (x). Далее представлены некоторые комбинации прав доступа:
1. 700 —владелец имеет полный доступ (чтение, запись, выполнение), группа и другие пользователи не имеют никаких прав.
2. 755 — владелец имеет полный доступ, остальные (группа и другие пользователи) могут только читать и выполнять.
3. 644 — владелец может читать и записывать, остальные могут только читать.
4. 600 —владелец может читать и записывать, остальные не имеют прав.
5. 777 —все пользователи (владелец, группа, остальные) могут читать, записывать и выполнять.
6. 664 — владелец и группа могут читать и записывать, остальные могут только читать.
7. 755 — владелец имеет полный доступ, а остальные могут читать и выполнять.
8. 775 — владелец и группа имеют полный доступ, остальные могут только читать и выполнять.

Эти комбинации позволяют настраивать доступ к файлам и директориям в зависимости от их предназначения и уровня конфиденциальности.

Несколько примеров для ознакомления:
1.Создание каталога "docs" в домашней директории пользователя:
mkdir ~/docs
2.Создание вложенного каталога:
mkdir ~/projects/2024
. Если "projects" уже существует, в нем будет создан каталог "2024". Иначе будет вызвана ошибка.
3.Создание вложенных каталогов с опцией `-p`:
mkdir -p ~/work/reports/2023
Если ни одного из каталогов "work" и "reports" нет, они будут созданы автоматически.
4.Переход в каталог в домашней директории:
cd ~/projects
5.Вывод полного пути к текущему каталогу:
pwd
6.Просмотр содержимого каталога (все файлы и каталоги):
ls ~/projects
7.Создание нескольких каталогов одновременно, а именно трех каталогов "frontend", "backend" и "2024" в "projects":
mkdir ~/projects/{frontend,backend,2024}
8.Создание каталога с пробелом в имени, а именно "new project" в "projects":
mkdir ~/projects/"new project"
9.Удаление каталога «2024», если он пустой:
rmdir ~/projects/2024
10.Удаление каталога «2024» с содержимым:
rm -r ~/projects/2024
11.Копирование каталога "2024" и все его содержимое в каталог "backup":
cp -r ~/projects/2024 ~/backup/2024_copy
15.Перемещение каталога "2024" в директорию "archives/old_projects":
mv ~/projects/2024 ~/archives/old_projects
16.Переименование каталога "2024" в "archive_2024":
mv ~/projects/2024 ~/projects/archive_2024
17.Создание скрытого каталога:
mkdir ~/.config_files
18.Показ скрытых файлов и каталогов:
ls -a ~/projects
19.Создание и изменение прав доступа к каталогу:
mkdir ~/secure_folder
chmod 700 ~/secure_folder
Создаёт каталог "secure_folder" и устанавливает права доступа только для владельца (чтение, запись, выполнение).

Решение:
1. mkdir ~/projects/2024 — будет создан вложенный каталог "2024" в "projects", если "projects" уже существует, таким образом, это верный ответ на вопрос задачи.
2. mkdir -p ~/projects/2024 — благодаря -p будут созданы и "projects", и "2024", если каталоги отсутствуют.
3. mkdir ~/projects && mkdir ~/2024 — сначала будет создан "projects", а затем будет создание "2024" в домашнем каталоге, что неправильно по условию задачи.
4. mkdir projects/2024 — это создание каталогов в текущем рабочем каталоге, а не в домашней директории.

Методика оценки заданий: если выбран верный ответ, участник получает 6 баллов.
Если выбран неверный ответ, участник получает 0 баллов.

Описание возможных трудностей при подготовке: Для облегчения работы с директориями, каталогами и файлами в Unix-подобных системах важно запомнить команды из теоретической вставки, которые помогут глубже разобраться в управлении файловой системой и автоматизации процессов.

Разбор типичных ошибок:
Обратите внимание на нюансы других вариантов ответа:

- mkdir -p ~/projects/2024 — благодаря -p будут созданы и "projects", и "2024", если каталоги отсутствуют;
- mkdir ~/projects && mkdir ~/2024 — сначала будет создан "projects", а затем будет создание "2024" в домашнем каталоге, что неправильно по условию задачи;
- mkdir projects/2024 — это создание каталогов в текущем рабочем каталоге, а не в домашней директории.

3. Сеть фирмы содержит подсеть с адресом 192.168.1.0/24. Администратору требуется разделить эту подсеть на две равные отдельные подсети. Какой из следующих адресов может быть адресом сети для одной из новых подсетей?

   А) 192.168.1.128/25
   Б) 192.168.1.128/27
   В) 192.168.1.0/26
   Г) 192.168.1.192/26

Ответ: А

Теория:
Каждая IP-сеть имеет маску сети, которая определяет, какая часть адреса относится к сети, какая — к устройству.
CIDR-нотация (Classless Inter-Domain Routing): формат записи IP-адреса с маской сети.
Рассмотрим пример: 192.168.1.0/24, где /24 обозначает, что первые 24 бита относятся к сети, а остальные — к устройствам.
Чтобы произвести разделение сети на две равные части, нужно сделать следующее:
Для сети 192.168.1.0/24 это приведёт к двум подсетям:
1. 192.168.1.0/25 (адреса от 192.168.1.0 до 192.168.1.127)
2. 192.168.1.128/25 (адреса от 192.168.1.128 до 192.168.1.255)

Решение:
Изначальная сеть — 192.168.1.0/24. Требуется разделить её на две равные сети. Это возможно с помощью маски /25, важно отметить, что подсети равны:

- 192.168.1.0/25 (192.168.1.0 – 192.168.1.127) – первая подсеть
- 192.168.1.128/25 (192.168.1.128 – 192.168.1.255) – вторая подсеть

Таким образом, верный ответ – А.

Рассмотрим еще несколько примеров разбиения на равные подсети, а также определим какие из следующих адресов могут быть адресом из новой подсети.

Пример на разделение на 4 подсети:
Рассмотрим сеть 192.168.10.0/24. Необходимо разделить её на четыре равные подсети. Какой из следующих адресов может быть адресом сети для одной из новых подсетей?

1. 192.168.10.64/27
2. 192.168.10.128/26
3. 192.168.10.192/25
4. 192.168.10.0/29

Решение:
Чтобы разделить изначальную сеть на четыре равные части, нужно произвести изменение маски на 2 бита: /24 → /26 (каждый шаг на бит в маске удваивает количество подсетей).
Получаем 4 подсети:

- 192.168.10.0/26 (адреса от 192.168.10.0 до 192.168.10.63)
- 192.168.10.64/26 (адреса от 192.168.10.64 до 192.168.10.127)
- 192.168.10.128/26 (адреса от 192.168.10.128 до 192.168.10.191)
- 192.168.10.192/26 (адреса от 192.168.10.192 до 192.168.10.255)

Таким образом получаем, что подходящий ответ: 2. 192.168.10.128/26.

Пример на разделение на 8 равных подсетей:

Рассмотрим сеть 10.0.0.0/16. Необходимо разделить её на восемь равных подсетей. Какой из следующих адресов может быть адресом одной из новых подсетей?

Варианты:
1. 10.0.32.0/13
2. 10.0.64.0/14
3. 10.0.128.0/18
4. 10.0.192.0/19

Решение:
Чтобы разделить на восемь равных подсетей сеть 10.0.0.0/16, нужно /16 → /19.
Тогда получим 8 равных подсетей:
10.0.0.0/19 (адреса от 10.0.0.0 до 10.0.31.255)
10.0.32.0/19 (адреса от 10.0.32.0 до 10.0.63.255)
10.0.64.0/19 (адреса от 10.0.64.0 до 10.0.95.255)
10.0.96.0/19 (адреса от 10.0.96.0 до 10.0.127.255)
10.0.128.0/19 (адреса от 10.0.128.0 до 10.0.159.255)
10.0.160.0/19 (адреса от 10.0.160.0 до 10.0.191.255)
10.0.192.0/19 (адреса от 10.0.192.0 до 10.0.223.255)
10.0.224.0/19 (адреса от 10.0.224.0 до 10.0.255.255)

Таким образом получаем, что подходящий ответ: 10.0.192.0/19.
Методика оценки заданий: если выбран верный ответ, участник получает 6 баллов.
Если выбран неверный ответ, участник получает 0 баллов.

Описание возможных трудностей при подготовке: необходимо повторить CIDR-нотацию (Classless Inter-Domain Routing): формат записи IP-адреса с маской сети, а также правила перевода из двоичной системы в десятичную.

Разбор типичных ошибок: типичные ошибки могут возникнуть, если не повторить правила работы с масками.

4. Предложена рекурсивная функция:

```
def s_fun(n):
        if n == 0:
                return 0
        else:
                return n % 10 + s_fun(n // 10)
```

Какой результат будет получен после вызова `s_fun(253)`?
А) 5
Б) 12
В) 10
Г) 7

Ответ: В
Теория:
Рекурсивные функции схематично можно представить в следующем виде:

```
def recursive_function(n):
   if условие_завершения:
      return результат
   else:
      return часть_решения + recursive_function(оставшаяся_часть_задачи)
```

n % 10 — позволяет выделить последнюю цифру числа, а именно вычисляет остаток числа от деления на 10.
n // 10 — удаляет последнюю цифру из числа, что даёт новую задачу с меньшим числом, а именно выполняет целочисленное деление на 10.

Решение:
Краткий вариант:
1. Первый вызов: 253 % 10 = 3, затем рекурсивно будет вызвана функция s_fun(25)
2. Второй вызов: 25 % 10 = 5, затем рекурсивно будет вызвана функция s_fun(2)
3. Третий вызов: 2 % 10 = 2, затем рекурсивно будет вызвана функция s_fun(0)
Подробное решение:
Для вызова функции s_fun(253):
1. s_fun(253) = 253 % 10 + s_fun(253 // 10), где:
   253 % 10 = 3
    s_fun(25)
2. s_fun(25) = 25 % 10 + s_fun(25 // 10), где:
    25 % 10 = 5
   s_fun(2)
3. s_fun(2) = 2 % 10 + s_fun(2 // 10) где:
   2 % 10 = 2
   s_fun(0)
4. s_fun(0) возвращает 0, так как условие завершения выполнено.
Таким образом, результат: 3 + 5 + 2 = 10.

Методика оценки заданий: Если выбран верный ответ, участник получает 6 баллов.
Если выбран неверный ответ, участник получает 0 баллов.

Описание возможных трудностей при подготовке: необходимо повторить изученный материал по программированию в рамках школьного курса. Важно знать: операторы, циклы, рекурсивные функции.

Разбор типичных ошибок: типичные ошибки могут возникнуть из-за невнимательности, поэтому при решении задания аккуратно перепроверьте полученные результаты и сопоставьте их с вопросом задачи.

5. Ученики 11 А, 11 Б, 11 В, 11 Г посещают различные кружки после уроков. Известно, что 35 человек посещают кружок программирования; 30 человек посещают кружок математики; 25 человек посещают кружок робототехники; 20 человек посещают кружок по физике; 10 учеников посещают кружки программирования и математики; 8 учеников посещают кружки программирования и робототехники; 7 учеников посещают кружки программирования и физики; 9 учеников посещают кружки математики и робототехники; 6 учеников посещают кружки математики и физики; 5 учеников, которые посещают кружки робототехники и физики; 4 ученика посещают кружки программирования, математики и робототехники; 3 ученика посещают кружки программирования, математики и физики; 2 ученика посещают кружки программирования, робототехники и физики; 2 ученика посещают кружки математики, робототехники и физики; 1 ученик посещает все четыре кружка.
   Сколько всего различных учеников посещают хотя бы один кружок?

   А) 80
   Б) 65
   В) 70
   Г) 75

   Ответ: Г.

Теория:
Для решения задач на множества важно знать формулу включений и исключений для n конечных множеств:

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{i} |A_i| - \sum_{i<j} |A_i \cap A_j| + \sum_{i<j<k} |A_i \cap A_j \cap A_k| - \ldots + (-1)^{n-1} |A_1 \cap A_2 \cap \ldots \cap A_n|.$$

Для любых двух конечных множеств A и B справедливо равенство:
$|A \cup B| = |A| + |B| - |A \cap B|$.
В сумме $|A|+|B|$ пересечение $A \cap B$ учтено дважды, поэтому необходимо вычесть $|A \cap B|$.
Для любых трех конечных множеств A, B и C справедливо равенство:
$|A \cup B \cup C| = |A| + |B| + |C| + |D| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$.
Для четырех множеств:
$|A \cup B \cup C \cup D| = |A| + |B| + |C| + |D| - |A \cap B| - |A \cap C| - |B \cap C| - |A \cap D| - |B \cap D| - |C \cap D| + |A \cap B \cap C| + |A \cap B \cap D| + |A \cap C \cap D| + |B \cap C \cap D| - |A \cap B \cap C \cap D|$.

Решение:
По условию задачи дано 4 множества:
Выделим по пунктам информацию про множества:

- 35 человек посещают кружок программирования,
- 30 человек посещают кружок математики,
- 25 человек посещают кружок робототехники,
- 20 человек посещают кружок по физике,
- 10 учеников посещают кружки программирования и математики,
- 8 учеников посещают кружки программирования и робототехники,
- 7 учеников посещают кружки программирования и физики,
- 9 учеников посещают кружки математики и робототехники,
- 6 учеников посещают кружки математики и физики,
- 5 учеников посещают кружки робототехники и физики,
- 4 ученика посещают кружки программирования, математики и робототехники,

- 3 ученика посещают кружки программирования, математики и физики,
- 2 ученика посещают кружки программирования, робототехники и физики,
- 2 ученика посещают кружки математики, робототехники и физики,
- 1 ученик посещает все четыре кружка.

Применим формулу для 4-х множеств $|A \cup B \cup C \cup D| = |A| + |B| + |C| + |D| - |A \cap B| - |A \cap C| - |B \cap C| - |A \cap D| - |B \cap D| - |C \cap D| + |A \cap B \cap C| + |A \cap B \cap D| + |A \cap C \cap D| + |B \cap C \cap D| - |A \cap B \cap C \cap D|$.

Подставим числовые значения:

$35 + 30 + 25 + 20 - (10 + 8 + 7 + 9 + 6 + 5) + (4 + 3 + 2 + 2) - 1 = 75$

Методика оценки заданий: Если выбран верный ответ, участник получает 6 баллов.
Если выбран неверный ответ, участник получает 0 баллов.

Описание возможных трудностей при подготовке:
Используйте диаграммы Эйлера/Венна для визуализации пересечений.

Разбор типичных ошибок:
1. Неправильный учёт пересечений множеств:
2. Игнорирование тройных и четверных пересечений:
3. Неправильное применение формулы включений и исключений:
4. Ошибки при расчёте пересечений:
5. Игнорирование общих множеств:
6. Учёт только части данных из задачи:

Как избежать ошибок:

- Выписывать все множества и пересечения.
- Следовать формуле включений и исключений.
- Перепроверять все расчёты.
- Применять визуализацию при решении задачи.