# Практическая работа №9 «Классы и объекты»

## Постановка задачи

### Часть 1.

1. Реализовать (в отдельном файле) определение нового класса/типа данных (закрытые атрибуты, свойства, конструкторы, инициализация и вывод атрибутов). Необходимо реализовать не менее 3 конструкторов: без параметров, с параметрами и конструктор копирования.
2. Для демонстрации работы с объектами написать главную функцию, в которой создаются объекты класса и выводится информация, которая содержится в атрибутах.
3. Написать функцию, выполняющую указанное в варианте действие. Рассмотреть два варианта реализации функции:

1) статическую функцию;

2) метод класса.

В основной программе продемонстрировать работу функций, объяснить разницу между статической и нестатической функциями.

1. Используя статическую компоненту класса подсчитать количество созданных в программе объектов.

#### Варианты:

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название класса** | **Атрибуты (поля)** | **Методы** |
| 1 | Rectangle | double length,  double width  (стороны прямоугольника) | Увеличить объекта Rectangle в N (int) раз. Учесть, что прямоугольник не может иметь стороны размером большим 46 340,9499 и меньшим 0,0001.  Результат должен быть типа Rectangle. |
| 2 | DialClock | int hours,  int minutes  (часы и минуты) | Вычислить угол между часовой и минутной стрелкой (в градусах) объекта DialClock.  Результат должен быть типа double, допустимо округление до 0,0001. |
| 3 | Car | double fuelFlow,  double fuelVolume  (расход топлива в литрах на 100 км и объём топлива в автомобиле в литрах) | Вычислить оставшийся запас хода объекта Car в километрах исходя из текущего объёма и расхода топлива. Учитывать единицы измерения и разумные ограничения значений атрибутов.  Результат должен быть типа double, допустимо округление до 0,001. |
| 4 | ChessboardCell | int horizontal,  int vertical  (горизонтальная и вертикальная координата клетки на шахматной доске, значения от 1 до 8) | Определить, имеют ли два объекта ChessboardCell одинаковый цвет на шахматной доске.  Результат должен быть типа bool. |
| 5 | Pokemon | int attack,  int defense,  int stamina  (значения атаки, защиты и выносливости покемона) | Увеличить характеристики покемона на заданные значения для каждой характеристики.  Учесть, что attack может принимать значения от 17 до 414, defense – от 32 до 396, stamina – от 1 до 496 единиц.  Результат должен быть типа Pokemon. |
| 6 | GeoCoordinates | double latitude,  double longitude  (широта и долгота координаты географической точки) | Вычислить расстояние (в км) между двумя географическими точками. Воспользуйтесь формулой гаверсинуса для вычисления расстояния между двумя точками на сфере.  Необходимо учитывать ограничения на допустимые десятичные значения широты и долготы.  Результат должен быть типа double, допустимо округление до 0,001. |
| 7 | CalendarDate | int day,  int month,  int year  (день, месяц, год) | Определения. является ли год даты високосным.  Учитывать, что в разные месяцы допускается разный диапазон допустимых значений для дня, а также максимальное количество дней в феврале у високосного года – 29.  Результат должен быть типа bool. |
| 8 | Discipline | string name,  int contactHours,  int selfHours  (название дисциплины, часы аудиторной и самостоятельной работы) | Вычислить количество зачётных единиц по дисциплине (1 з.е. = 38 часов).  Результат должен быть типа int. |
| 9 | Student | string name,  int age,  double gpa  (имя, возраст и Grade Point Average – усреднённый бал всех итоговых оценок студента) | Сравнить двух студентов по возрасту и gpa.  Результат должен быть типа string в формате «{Имя студента 1} младше/старше/ровесник {Имя студента 2}. GPA {Имя студента 1} выше/ниже/равен GPA {Имя студента 2}» |
| 10 | Weather | double temperature,  int humidity,  int pressure  (температура в градусах Цельсия, влажность в %, давление в мм рт. ст.) | Вычислить точку росы по заданным параметрам температуры и относительной влажности объекта. Точку росы можно найти по формуле:  , где  a – константа = 17,27;  b – константа = 237,7;  Т – temperature;  Rh – humidity.  Результат должен быть типа double, допустимо округление до 0,0001. |
| 11 | Runner | double speed,  double distance  (средняя скорость в км/ч, расстояние в км) | Вычислить время преодоления дистанции в часах.  Результат должен быть типа double, допустимо округление до 0,01. |
| 12 | CarParking | int numSlots,  int numCars  (количество парковочных мест и автомобилей на парковке) | Вычислить загруженность парковки в процентах.  Результат должен быть типа double в диапазоне от 0 до 100, допустимо округление до 0,01. |
| 13 | Post | int numViews,  int numComments,  int numReactions  (количество просмотров, комментариев и реакций на пост в сообществе социальной сети) | Вычислить коэффициент вовлечённости подписчиков сообщества по данному посту, если всего подписчиков у сообщества 1000.  Результат должен быть типа double в диапазоне от 0 до 100, допустимо округление до 0,01. |
| 14 | Dish | double proteins,  double fats,  double carbohydrates  (количество белков, жиров и углеводов на 100 г некоторого блюда) | Вычислить калорийность блюда по формуле 4 \* proteins + 9 \* fats +  4 \* carbohydrates.  Результат должен быть типа double, допустимо округление до 0,01. |
| 15 | Mark | string name,  int mark  (название дисциплины и оценка от 0 до 10 включительно) | Определить оценку в пятибальной шкале согласно системе перевода, используемой в НИУ ВШЭ.  Результат должен быть типа string. |

### Часть 2.

1. Добавить к реализованному классу указанные в варианте перегруженные операции.
2. Добавить метод public override bool Equals(object obj) {} для сравнения двух объектов реализованного класса (без этого метода не будут работать unit-тесты).
3. Написать демонстрационную программу, в которой создаются объекты пользовательских классов и выполняются указанные операции.

#### Варианты:

|  |  |  |
| --- | --- | --- |
| **№** | **Название класса** | **Методы** |
| 1 | Rectangle | Унарные операции:  ++ добавление единицы к сторонам прямоугольника (учитывать ограничения из 1 части);  -- вычитание единицы из сторон прямоугольника (учитывать ограничения из 1 части).  Операции приведения типа:  double (явная) – результатом является значение площади описанной вокруг прямоугольника окружности, допускается округление до 0,0001;  bool (неявная) – результатом является true, если прямоугольник является квадратом, и false в противном случае.  Бинарные операции:  + Rectangle r, вещественное число (лево- и право- сторонние операции увеличения сторон прямоугольника на заданное число). Результат должен быть типа Rectangle;  - Rectangle r, вещественное число (лево- и право- сторонние операции уменьшения сторон прямоугольника на заданное число). Результат должен быть типа Rectangle. |
| 2 | DialClock | Унарные операции:  ++ добавление минуты к объекту типа DialClock (учесть, что минут может быть от 0 до 59, а часов – от 0 до 23 включительно);  -- вычитание минуты из объекта типа DialClock (учесть те же ограничения).  Операции приведения типа:  bool (явная) – результатом является true, если угол между часовой и минутной стрелкой кратен 2,5, иначе – false;  int (неявная) – результатом является количество пройденных делений минутной стрелкой от начального положения обеих стрелок (когда обе стрелки указывают на число 12 циферблата часов) до текущего включительно.  Бинарные операции:  + DialClock dc, целое число (минуты) (лево- и право- сторонние операции). Результат должен быть типа DialClock;  - DialClock dc, целое число (минуты) (лево- и право- сторонние операции). Результат должен быть типа DialClock. |
| 3 | Car | Унарные операции:  ++ увеличение расхода топлива автомобиля на 0,1 л / 100 км;  -- уменьшение топлива в баке автомобиля на 1 л (топлива в баке не может быть меньше 0).  Операции приведения типа:  bool (явная) – результатом является true, если автомобиль сможет доехать до заправки (до заправки ровно 100 км), а в баке в момент заправки останется не меньше 5 л топлива, иначе – false;  double (неявная) – результатом является количество сотен километров до заправки, чтобы в момент заправки в баке осталось ровно 5 л топлива. Если в момент расчёта в баке меньше 5 л топлива, значит результатом операции будет число -1.  Бинарные операции:  + Car c, вещественное число (количество литров топлива) (левосторонняя операция добавления некоторого количества литров топлива в топливный бак автомобиля). Результат должен быть типа Car;  + вещественное число, Car c (правосторонняя операция, уменьшается расход топлива на заданное число). Результат должен быть типа Car;  == Car c1, Car c2 – автомобили имеют равные возможности, если равны их атрибуты;  != Car c1, Car c2 – автомобили не равносильны, если не равны их атрибуты. |
| 4 | ChessboardCell | Унарные операции:  ++ увеличение обеих координат клетки на 1;  ! поменять координаты клетки относительно главной диагонали. Если клетка расположена на главной диагонали, то координаты не меняются.  Операции приведения типа:  int (явная) – результатом является сумма координат клетки шахматной доски;  string (неявная) – результатом является цвет клетки на шахматной доске («чёрная» или «белая»).  Бинарные операции:  == ChessboardCell cc1, ChessboardCell cc2 – результатом является true, если фигура коня может переместиться между двумя клетками за один ход, и false – в противном случае.  != ChessboardCell cc1, ChessboardCell cc2 – результатом результатом является true, если фигуры на заданных клетках находятся на разных вертикалях доски, и false – в противном случае. |
| 5 | Pokemon | Унарные операции:  ~ возвращает результат вычисления боевой мощи покемона CP (используйте упрощённую формулу Pokemon GO IV Formula, где STA – stamina, ATK – attack, DEF – defense), допустимо округление до 0,01;  -- уменьшение выносливости покемона на 1.  Операции приведения типа:  int (явная) – результатом является сумма всех характеристик покемона;  double (неявная) – результатом является среднее значение характеристик покемона, округлённое до 2 знаков после запятой.  Бинарные операции:  >> Pokemon p, целое число – результатом является объект p, у которого увеличивается выносливость на заданное целое число единиц (правосторонняя операция).  > Pokemon p, целое число – результатом является объект p, у которого увеличивается защита на заданное целое число единиц (правосторонняя операция).  < Pokemon p, целое число – результатом является объект p, у которого увеличивается атака на заданное целое число единиц (правосторонняя операция). |
| 6 | GeoCoordinates | Унарные операции:  ++ увеличение широты и долготы объекта на 0,01;  - широта и долгота точки меняет знак значения на противоположный, абсолютное значение не меняется.  Операции приведения типа:  bool (явная) – результатом является true, если точка располагается на экваторе, иначе – false;  string (неявная) – результатом является определение типа долготы точки: «Восточная долгота» / «Западная долгота» / «Нулевой меридиан».  Бинарные операции:  == GeoCoordinates gc1, GeoCoordinates gc2 – результатом является true, если обе точки находятся на одной параллели, и false – в противном случае.  != GeoCoordinates gc1, GeoCoordinates gc2– результатом является true, если точки находятся на разных меридианах, и false – в противном случае. |
| 7 | CalendarDate | Унарные операции:  true – проверяется, является ли указанная дата будущей или равной текущей;  false – проверяется, является ли указанная дата прошлой, относительно текущей.  Операции приведения типа:  int (явная) – результатом является номер квартала года для текущей даты;  string (неявная) – результатом является сформированное строковое представление даты в формате «ДД.ММ.ГГГГ».  Бинарные операции:  + CalendarDate cd, int days – (правосторонняя операция) результатом является новая дата относительно cd, у которой увеличивается значение дня. Учесть ограничения на количество дней в месяце и месяцев в году, а также количество дней может быть отрицательным;  >> CalendarDate cd, int months – (правосторонняя операция) результатом является новая дата относительно cd со сдвигом на указанное количество месяцев (может быть задано отрицательным числом). Все ограничения на дату должны быть также соблюдены. |
| 8 | Discipline | Унарные операции:  ! определение процентного соотношения самостоятельной работы к общему количеству часов, выделенных на дисциплину, результат – вещественное число от 0 до 100 (без символа %);  ++ увеличение аудиторного количества часов на 2, учесть, что общее количество часов, выделенных на дисциплину, не должно меняться (к аудиторным часам добавляем, из самостоятельных – вычитаем).  Операции приведения типа:  double (явная) – результатом является доля часов аудиторной работы по дисциплине (число от 0 до 1 включительно);  int (неявная) – результатом является количество аудиторных занятий выделенных на дисциплину (одна пара – это 2 академических часа).  Бинарные операции:  >= Discipline d1, Discipline d2 – сравниваются трудоёмкости на две дисциплины.  <= Discipline d1, Discipline d2 – сравниваются трудоёмкости на две дисциплины. |
| 9 | Student | Унарные операции:  ~ приведение имени студента к формату: первая буква заглавная, остальные строчные;  ++ увеличить возраст студента на 1.  Операции приведения типа:  int (явная) – результатом является номер курса, на котором обучается студент (если студенту 18 лет, значит, он обучается на первом курсе, если студенту больше 22 лет – возвращать -1, как флаг невозможности точного определения номера курса);  bool (неявная) – результатом является true, если gpa < 6 (студент скорее всего имеет удовлетворительные оценки), иначе – false.  Бинарные операции:  % Student s, string newName – результатом является новый студент, у которого возраст и gpa идентичен s, но имя другое;  - Student s, double d – (правосторонняя операция) результатом является тот же студент, но с уменьшенным gpa на заданное число d. Учитывать, что gpa не может быть меньше 0. |
| 10 | Weather | Унарные операции:  - получение нового объекта погоды с температурой обратной по знаку с исходной;  ! результатом является true, если влажность выше 80%, иначе – false.  Операции приведения типа:  bool (явная) – результатом является true, если давление выше нормы в 760 мм рт. ст., иначе – false;  double (неявная) – результатом является ощущаемая температура при заданной температуре воздуха и влажности (humindex), допустимо округление до 0,01. Можно использовать формулу относительно данных в погоде параметров температуры и относительной влажности, либо через температуру и точку росы.  Бинарные операции:  - Weather w, double d – (правосторонная операция) результатом является новый объект погоды, в котором температура уменьшается на заданное число (может быть задано отрицательное значение);  \* Weather w, double percent – (правосторонняя операция) результатом является новый объект погоды, в котором параметры изменяются на заданное число процентов. Например, w \* 12,5 значит, что все параметры увеличиваются на 12,5% (если начальная температура была равна 20, значит, у нового объекта температура будет равна 20 \* 1,125 = 22,5). |
| 11 | Runner | Унарные операции:  ++ увеличение расстояния на 0,1 км;  -- уменьшение скорости на 0,05 км/ч.  Операции приведения типа:  double (явная) – результатом является значение, на которое нужно увеличить скорость, чтобы время прохождения дистанции сократилось на 5%;  string (неявная) – результатом является строка со временем преодоления дистанции при заданной скорости в формате «ЧЧ:ММ:СС».  Бинарные операции:  - Runner r1, Runner r2 – два бегуна стартуют одновременно навстречу друг другу, находясь на расстоянии 15 км друг от друга. Определить на каком расстоянии произойдёт встреча двух бегунов, если это возможно с учётом расстояния и скорости каждого бегуна. Если бегуны не встретятся, то результат операции -1;  ^ Runner r1, double sp – результатом является новый объект бегуна, у которого увеличена скорость на заданное число. |
| 12 | CarParking | Унарные операции:  ++ увеличение количества автомобилей на парковке на 1;  -- уменьшение количества автомобилей на парковке на 1.  Операции приведения типа:  int (явная) – количество автомобилей, которых не хватает до загруженности парковки на 80%;  bool (неявная) – результатом является true, если на парковке есть свободные места, иначе – false.  Бинарные операции:  + CarParking cp1, CarParking cp2 – объединение двух парковок в одну.  > CarParking cp1, CarParking cp2 – результатом является true, если на парковке cp1 загруженность ниже и общее количество мест больше, чем на парковке cp2, и false – в противном случае.  < CarParking cp1, CarParking cp2 – результатом является true, если на парковке cp1 загруженность выше и общее количество мест меньше, чем на парковке cp2, и false – в противном случае. |
| 13 | Post | Унарные операции:  ! увеличить количество реакций на 1;  ++ увеличить количество просмотров на 1.  Операции приведения типа:  bool (явная) – результатом является true, если у поста есть хотя бы один комментарий или одна реакция при ненулевом количестве просмотров, иначе – false;  double (неявная) – результатом является число – процент охвата (какой процент подписчиков сообщества посмотрел публикацию), если подписчиков у сообщества 1000, допустимо округление до 0,01.  Бинарные операции:  == Post p1, Post p2 – результатом является true, если оба поста имеют одинаковые показатели, и false – в противном случае.  != Post p1, Post p2 – результатом является true, если хотя бы один показатель у постов отличается, и false – в противном случае. |
| 14 | Dish | Унарные операции:  ++ увеличить количество белков, жиров и углеводов на 1;  ~ вычислить процент калорийности блюда от суточной нормы в 2000, допускается округление до 0,01.  Операции приведения типа:  bool (явная) – результатом является true, если соотношение белков, жиров и углеводов в блюде является идеальным (идеал – это соотношение 3:3:4, или 30%, 30% и 40% соответственно);  string (неявная) – результатом является строка в формате «Белков – {proteins} г., жиров – {fats} г., углеводов – {carbohydrates} г.».  Бинарные операции:  \* Dish d, double portion – (право- и лево- сторонняя операция) результатом является целое число – общая калорийность блюда. Portion – это порция блюда в граммах;  > Dish d1, Dish d2 – результатом является true, если калорийность первого блюда выше калорийности второго, и false – в противном случае;  < Dish d1, Dish d2 – результатом является true, если калорийность первого блюда ниже калорийности второго, и false – в противном случае. |
| 15 | Mark | Унарные операции:  ! изменить название дисциплины так, чтобы все буквы были заглавными;  - обнулить оценку по дисциплине.  Операции приведения типа:  int (явная) – количество буквенных символов в названии дисциплины;  bool (неявная) – результат true, если оценка по дисциплине не меньше 4.  Бинарные операции:  / Mark m, string newName – замена названия дисциплины на новое.  >= Mark m1, Mark m2 – результатом является true, если первая оценка не меньше второй, и false – в противном случае;  <= Mark m1, Mark m2 – результатом является true, если первая оценка не больше второй, и false – в противном случае. |

### Часть 3

1. Реализовать класс-коллекцию (в отдельном файле), полем которого является **одномерный массив** (**не использовать стандартные коллекции C#!)** из элементов заданного в варианте типа. Например, для класса Money нужно создать класс MoneyArray следующим образом:

class MoneyArray

{

Money[] arr;//одномерный массив элементов типа Money

. . . .

}

В классе реализовать:

* конструктор без параметров;
* конструктор с параметрами, заполняющий элементы случайными значениями;
* конструктор с параметрами, позволяющий заполнить массив элементами, заданными пользователем с клавиатуры;
* конструктор копирования, позволяющий создать копию коллекции, которая передается в конструктор как параметр, д.б. реализовано глубокое копирование.
* метод для просмотра элементов массива.

1. Реализовать индексатор для доступа к элементам коллекции. Предусмотреть проверку при выходе индекса за пределы массива и обработку этой ситуации с помощью **исключительных ситуаций.**
2. Написать демонстрационную программу, позволяющую создать массив разными способами (ручной ввод и случайная генерация) и распечатать элементы массива. Создать новую коллекцию на основе существующей, показать, что выполнено глубокое копирование.
3. В демонстрационной программе показать 4 варианта работы индексатора (запись объекта и получение объекта с существующим индексом, запись объекта и получение объекта с несуществующим индексом).
4. Написать функцию **в классе Program** для выполнения указанного в варианте задания (**использовать индексатор** и, если необходимо, перегрузить нужные для выполнения задачи операции), т.е. функция должна перебирать массив arr и обрабатывать его элементы, используя перегруженные операции пользовательского класса.
5. Подсчитать количество созданных объектов и созданных коллекций.

#### Варианты

|  |  |  |
| --- | --- | --- |
| **№** | **Название класса** | **Задание** |
| 1 | RectangleArray | Найти среднее арифметическое площадей описанных вокруг прямоугольников окружностей, допускается округление до 0,0001. |
| 2 | DialClockArray | Найти объект DialClock с максимальным углом между часовой и минутной стрелкой. |
| 3 | CarArray | Найти автомобиль с наименьшим запасом хода. |
| 4 | ChessboardCellArray | Определить самую близкую клетку к левому нижнему углу шахматной доски (началу координат шахматной доски – это когда по вертикали и горизонтали значения координат равны 1). |
| 5 | PokemonArray | Найти моду выносливости покемонов. |
| 6 | GeoCoordinatesArray | Найти ближайшую географическую точку к «Острову Ноль» (вымышленное название географической точки с 0 по широте и долготе). |
| 7 | CalendarDateArray | Найти максимальное количество дат одного года в массиве дат. |
| 8 | DisciplineArray | Найти средневзвешенное значение зачётных единиц по всем дисциплинам в массиве |
| 9 | StudentArray | Найти самого старшего студента с gpa > 8. Если таких студентов нет – возвращать -1. |
| 10 | WeatherArray | Найти амплитуду температур из массива данных о погоде. |
| 11 | RunnerArray | Отсортировать массив бегунов по расстоянию от большего к меньшему, затем если расстояние совпадает, то по времени. |
| 12 | CarParkingArray | Найти менее загруженную парковку и вывести количество оставшихся свободных мест на ней. |
| 13 | PostArray | Найти общий коэффициент вовлечённости по постам одного сообщества (в заданном массиве), если подписчиков 1000. |
| 14 | DishArray | Найти наиболее калорийное блюдо. |
| 15 | MarkArray | Найти все дисциплины с оценкой выше средней. |

# Методические указания:

1. Операции ввода данных и вывода результата (пользовательский интерфейс) реализовать в отдельном классе.
2. При реализации конструктора использовать свойства для инициализации полей класса.
3. Для подсчета количества объектов использовать статическую переменную.
4. В индексаторе использовать исключения (throw new ArgumentException()).

# Содержание отчета:

1. Постановка задачи (общая и конкретного варианта)
2. Диаграмма классов для каждой части работы (из VS или Visio)
3. Код программы или ссылка на гитхаб с кодом
4. Код unit-тестов (или тесты в Excel) или ссылка на гитхаб с кодом тестов.
5. Анализ покрытия кода тестами (из VS или в Excel)