

Домашняя работа

Чучелина Таисия ИПО-22.24

Тема: “Переменные”

Вопрос 1

Что выведет следующий код:

```
1 stringname = "Tom";  
2 Console.WriteLine(Name);
```

Ответ:

Код выведет ошибку. Так как название переменных не совпадают.

Вопрос 2

Что выведет на консоль следующий код:

```
1 stringperson = "Tom";  
2 person = "Sam";  
3 Console.WriteLine(person);
```

Варианты ответов

- Том
- Sam
- person
- Программа завершит выполнение с ошибкой

Ответ: Программа завершит выполнение с ошибкой

Вопрос 3

Какие из следующих вариантов представляют корректное определение переменных:

stringperson = "Tom";

person = "Tom";

```
stringperson;
```

```
string"Tom";
```

Ответ: string person = "Tom";

Вопрос 4

Какие три основных компонента имеет переменная в языке C#?:

- класс, имя, метод
- тип, размер, область видимости
- имя, индекс, значение
- Тип, имя, значение

Ответ: Тип, имя, значение

Вопрос 5

В чём заключается различие между определением переменной и её инициализацией в C#?

- определение создаёт новую переменную в памяти, а инициализация её удаляет.
- определение задаёт начальное значение, а инициализация устанавливает тип переменной.
- Определение устанавливает тип и имя переменной, а инициализация задаёт начальное значение.
- определение и инициализация — это одно и то же действие.

Ответ: Определение устанавливает тип и имя переменной, а инициализация задаёт начальное значение.

Вопрос 6

Почему важно учитывать регистрозависимость при работе с переменными в C#? Приведите пример.

- Регистр важен для типов данных, а не для имён переменных.

- C# регистрозависимый язык, поэтому name и Name — разные переменные.
- В C# регистр не имеет значения для имён переменных.
- Имена переменных в C# должны быть записаны только строчными буквами.

Ответ: C# регистрозависимый язык, поэтому name и Name — разные переменные.

Пример:

```
int name = 5;  
int Name = 10;  
Console.WriteLine(name); // выводит 5  
Console.WriteLine(Name); // выводит 10
```

Вопрос 7

В чём состоит ключевое отличие константы от переменной в C# и как это отражается на их использовании в программе?

- Значение переменной фиксируется при определении и не может быть изменено.
- Константа может быть изменена в процессе работы программы, как и переменная.
- Переменные и константы в C# ничем не отличаются друг от друга.
- Константа инициализируется при определении и её значение нельзя изменить, в отличие от переменной.

Ответ: Константа инициализируется при определении и её значение нельзя изменить, в отличие от переменной.

Литералы

Вопрос 1

Какие виды литералов существуют и чем они отличаются друг от друга?

- Логические, целочисленные, вещественные, символьные, строковые и null.
- целые, дробные, текстовые, булевые и специальные.
- положительные, отрицательные, дробные, символьные и строковые.
- числовые, буквенные, логические, графические и пустые.

Ответ: Логические, целочисленные, вещественные, символьные, строковые и null.

Вопрос 2

В каких формах могут быть представлены вещественные литералы и как они интерпретируются?

- строковые литералы в двойных кавычках
- Вещественные числа с фиксированной запятой и в экспоненциальной форме MЕр
- целые числа в десятичной, шестнадцатеричной и двоичной форме
- символьные литералы в одинарных кавычках

Ответ: Вещественные числа с фиксированной запятой и в экспоненциальной

Базовые типы данных

Вопрос 1

Какие из нижеперечисленных НЕ являются встроенными типами языка C#?

- uint
- sbyte
- real
- int128
- object
- float64

Ответ: real, int128, float64

Вопрос 2

Какой тип данных языка C# будет представлять следующая переменная?

1 bool enabled = true;

Ответ: Тип данных bool. Если будет написано с пробелом.

Вопрос 3

Какой тип данных языка C# будет представлять следующая переменная?

1 varweight = 84.45f;

Ответ: Тип данных var. Если будет написано с пробелом.

Вопрос 4

Сколько байт занимает значение типа **uint**?

Ответ: 4 байта.

Вопрос 5

Какие из следующих вариантов представляют корректное определение переменных:

1 stringperson = "Tom";

2 varperson = "Tom";

3 varperson;

4 stringperson;

Ответ: string person = "Tom"; и var person = "Tom";

Вопрос 6

Какой системный тип соответствует базовому типу данных **int** в языке C# и сколько байт он занимает?

1. System.Int32, 4 байта
2. System.Single, 4 байта
3. System.UInt32, 8 байт
4. System.Int16, 2 байта

Ответ: System.Int32, 4 байта.

Вопрос 7

Какие суффиксы используются в C# для явного указания типа данных float и decimal при присвоении значений?

1. S/s — для float, D/d — для decimal

2. X/x — для float, Y/y — для decimal
3. F/f — для float, M/m — для decimal
4. L/l — для float, U/u — для decimal

Ответ: F/f — для float, M/m — для decimal

Вопрос 8

Чем отличается объявление переменной с использованием var от явного указания типа данных, например, int?

1. var и int — это синонимы для объявления целочисленных переменных.
2. При использовании var тип переменной определяется автоматически на основе присвоенного значения.
3. var используется для объявления переменных с типом string.
4. var позволяет объявлять переменные без указания типа и инициализации.

Ответ: При использовании var тип переменной определяется автоматически на основе присвоенного значения.

Консольный ввод-вывов

Вопрос 1

Как вывести на консоль значения нескольких переменных в одной строке с помощью интерполяции?

- Console.WriteLine("{name} {age} {height}");
- Console.WriteLine("Имя: " name " Возраст: " age " Рост: " height "м");
- Console.WriteLine("Имя: {name} Возраст: {age} Рост: {height}м");
- Console.Write(name, age, height);

Ответ: Console.WriteLine(\$"Имя: {name} Возраст: {age} Рост: {height}м");

Если поставить перед строкой знак \$

Вопрос 2

Что такое плейсхолдеры в контексте вывода данных на консоль и как они используются?

- Плейсхолдеры — это числа в фигурных скобках, которые заменяются значениями при выводе на консоль
- плейсхолдеры используются для создания пустых строк в выводе
- плейсхолдеры — это имена переменных, которые выводятся на консоль без изменений
- плейсхолдеры — это специальные символы для форматирования строк

Ответ: Плейсхолдеры — это числа в фигурных скобках, которые заменяются значениями при выводе на консоль.

Вопрос 3

В чём отличие метода Console.WriteLine() от Console.Write()?

1. Console.WriteLine() используется для ввода данных, а Console.Write() — для вывода.
2. Console.WriteLine() выводит информацию в виде таблицы, а Console.WriteLine() — в виде списка.
3. Console.WriteLine() не добавляет переход на следующую строку, а Console.WriteLine() добавляет.
4. Console.WriteLine() может выводить только строки, а Console.WriteLine() — любые данные.

Ответ: Console.WriteLine() не добавляет переход на следующую строку, а Console.WriteLine() добавляет.

Вопрос 4

Каким методом можно получить ввод с консоли и в каком виде он возвращается?

1. методом Console.ReadLine(), возвращается в виде числа.
2. методом Console.WriteLine(), возвращается в виде массива.
3. методом Convert.ToInt(), возвращается в виде строки.
4. Методом Console.ReadLine(), возвращается в виде строки.

Ответ: Методом Console.ReadLine(), возвращается в виде строки.

Вопрос 5

Какие методы предоставляет платформа .NET для преобразования строковых значений в числовые типы данных?

1. Convert.ToString(), Convert.ToInt(), Convert.ToChar()
2. Parse.ToInt(), Parse.ToDouble(), Parse.ToDecimal()
3. Convert.ToInt(), Convert.ToDouble(), Convert.ToDecimal()
4. Console.WriteLine(), Console.Write(), Console.ReadLine()

Ответ: Convert.ToInt(), Convert.ToDouble(), Convert.ToDecimal()

Операции

Вопрос 1

Есть следующий код:

```
1 intn1 = 2;  
2 intn2 = 5;  
3 intresult = n2 * 3 + 20 / 2 * n1--;
```

Используя приоритеты операций, разложите выражение $\text{intresult} = \text{n2} * 3 + 20 / 2 * \text{n1}--$ по шагам.

Ответ:

1. **Вычислить постфиксный оператор n1--:**
 - Постфиксный декремент: сначала используем текущее значение n1 (2) в выражении, затем уменьшаем n1 на 1 (до 1).
 - Значение для выражения: 2.
2. **Вычислить подвыражение с умножением/делением: n2 * 3**
 - $\text{n2} = 5$, умножить на 3: $5 * 3 = 15$
3. **Вычислить следующее подвыражение: 20 / 2**
 - $20 / 2 = 10$ (целое деление)
4. **Вычислить умножение: 20 / 2 * n1--**
 - Теперь берём результат из шага 3 (10) и умножаем на значение из шага 1 (2): $10 * 2 = 20$
5. **Сложить результаты: n2 * 3 + (20 / 2 * n1--)**

- $15 + 20 = 35$

6. Присвоить результат:

- $\text{result} = 35$
- После выполнения $n1 = 1$ (из-за декремента).

Вопрос 2

Есть следующий код:

```
1 intnum1 = 4;
2 intnum2 = 5;
3 intnum3 = 15;
4 intnum4 = 10;
5 intnum5 = 5;
6 intresult = 12;
7
8 result += num1 * num2 + num3 % num4 / num5;
```

Используя приоритеты операций, разложите выражение $\text{result} += \text{num1} * \text{num2} + \text{num3} \% \text{num4} / \text{num5}$ по шагам.

Ответ:

1. Вычислить: $\text{num1} * \text{num2}$

- $\text{num1} = 4, \text{num2} = 5$
- Результат: $4 * 5 = 20$

2. Вычислить: $\text{num3} \% \text{num4}$:

- $\text{num3} = 15, \text{num4} = 10$
- Остаток от деления: $15 \% 10 = 5$

3. Вычислить: $\text{num3} \% \text{num4} / \text{num5}$:

- Результат: $5 / 5 = 1$

4. Вычислить: $\text{num1} * \text{num2} + (\text{num3} \% \text{num4} / \text{num5})$

- $20 + 1 = 21$

5. Выполнить оператор $+=$:

- $\text{result} = \text{result} + 21$
- $\text{result} = 12 + 21 = 33$

Вопрос 3

Чему будет равна переменная z после выполнения следующего кода и почему?

1 intx = 8;
2 inty = 9;
3 intz = x++ + ++y;

Ответ: z = 18

Значение x++ — **8**

Значение ++y — **10**

После этого:

x = **9**

y = **10**

В результате выражение:

$z = 8 + 10; // = 18$

Постфиксный инкремент x++ использует старое значение в выражении, а увеличение происходит после вычисления

