Изпитна тема № 1: Програмиране

Основни понятия: програмиране, език за програмиране, алгоритъм, среда за разработка (IDE), компилация и интерпретация. Пресмятания, оператори, изрази. Условни конструкции. Логически изрази и оператори за сравнение. Вложени условни оператори. Цикли. Вложени цикли. Подпрограми (функции/методи), параметри, връщана стойност.

Програмирането е компютърна наука, занимаваща се с алгоритмите, методите и средствата за създаване на компютърни програми. Програмите представляват поредица от команди, които са написани на предварително избран език за програмиране като C#, Java, JavaScript, Python, Ruby, PHP, C, C++, Swift, Go или друг. За да бъдат написани тези команди трябва да се спазва синтаксиса и семантиката на езика, с който ще се работи.

Езиците за програмиране изкуствени езици, предназначени за задавани команди, които да бъдат прочетени, обработени и изпълнени от компютъра. Езиците за програмиране се разделят на два основни вида. Езици от ниско и езици от високо ниво. Езиците от ниско ниво управляват хардуера директно и изискват много усилия и голям брой команди за да се извърши определено действие. Пример за такъв език е “Assembler”, с който могат директно да бъдат написани команди, управляващи процесора. Езиците от високо ниво улесняват програмирането, поради улеснения им синтаксис. С тези езици могат да бъдат създадени от операционни системи до уебсайтове и приложни програми. Примери за такива езици са C, C++, C#, JavaScript и тн.

Алгоритмите представляват последователност от краен брой стъпки (инструкции, команди, указания, оператори), които при изпълнението си водят до решаване на съответната задача. В програмирането алгоритъмът е добре подредена колекция от еднозначни и ефективно изчислими операции, които след изпълнение дават резултат.

За да бъде улеснена работата на програмистите се използва интегрирана среда за разработка (IDE – Integrated Development Environment). Това представлява съвкупност от инструменти за разработване на софтуерни приложения. В средата на разработка се пише изходния код, след което той бива компилиран или интерпретиран, а програмите се изпълняват. Средите за разработка съдържат в себе си текстов редактор за писане на кода, компилатор или интерпретатор и дебъгер, с който може да се проследи изпълнението на програмата, което спомага за търсене на грешки. Тази съвкупност от инструменти позволява разработката на многокомпонентни софтуерни продукти, без да се напуска дадената среда за разработка, което подобрява ефективността на работа. За разработка на C# приложения най-често се използва Visual Studio.

Езиците за програмиране се разделят и в две категории, спрямо това по какъв начин се изпълнява програмният код.

Във всички езици за програмиране се използват оператори, чрез които се извършват някакви действия върху данните. Операторите позволяват обработка на различните типове данни и обекти. Те могат да приемат един, два или три операнда и връщат стойност като резултат. Операторите в C# представляват специални символи като „+“, „.“, „\*“ и други. Те могат да бъдат разделени в няколко различни категории:

* Аритметични – използват се за извършване на математически изчисления.
* Оператори за присвояване – използват се за присвояване на различни стойности на променливите.
* Оператори за сравнение – позволяват сравнение на два литерала и/или променливи
* Побитови оператори – използват се за извършване на операции върху двоичното представяне на числата.
* Оператори за преобразуване – позволяват преобразуването на данни от един тип в друг.

Следва списък с операторите, разделени по категории:

|  |  |
| --- | --- |
| **Категория** | **Оператори** |
| аритметични | **-, +, \*, /, %, ++, --** |
| логически | **&&, ||, !, ^** |
| побитови | **&, |, ^, ~, <<, >>** |
| за сравнение | **==, !=, >, <, >=, <=** |
| за присвояване | **=, +=, -=, \*=, /=, %=, &=, |=, ^=, <<=, >>=** |
| съединяване на символни низове | **+** |
| за работа с типове | **(type), as, is, typeof, sizeof** |
| други | **., new, (), [], ?:, ??** |

Според броя на аргументите операторите са три вида – едноаргументни (unary), двуаргументни (binary) и триаргументни(ternary).

|  |  |
| --- | --- |
| **Вид** | **Оператори** |
| едноаргументни (unary) | ^, ~, ++, -- и тн. |
| двуаргументни (binary) | **+, -, \*, / и тн.** |
| триаргументни(ternary). | **?:** |

Някои оператори имат различни форми. Например операторите „+“ и „-“ имат както едноаргументна, така и двуаргумента форма. Операторът за увеличаване с единица (increment) “++” добавя единица към стойността на променливата, съответно операторът “--" (decrement) изважда единица от стойността. Когато използваме операторите “++” и “—" като суфикс (поставяме операторът непосредствено пред променливата) първо се пресмята новата стойност, а после се връща резултата и програмата продължава с решението на израза, докато при използването на операторите като постфикс (поста­вяме оператора непосредствено след променливата) първо се връща оригиналната стойност на операнда, а после се добавя или изважда единица към нея. Някои оператори имат приоритет над други. Например, както е в математиката, умножението има приоритет пред събирането. Операторите с по-висок приоритет се изчисляват преди тези с по-нисък. Операторът () служи за промяна на приоритета на операторите и се изчислява пръв, също както в математиката.

Логическите оператори приемат булеви стойности и връщат булев резултат (true или false). Основните булеви оператори са "И" (&&), "ИЛИ" (||), изключващо "ИЛИ" (^) и логическо отрицание (!).

Следва таблица с логическите оператори в C# и операциите, които те извършват:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **x** | **y** | **!x** | **x && y** | **x || y** | **x ^ y** |
| **true** | **true** | **false** | **true** | **true** | **false** |
| **true** | **false** | **false** | **false** | **true** | **true** |
| **false** | **true** | **true** | **false** | **true** | **true** |
| **false** | **false** | **true** | **false** | **false** | **false** |

Побитов оператор (bitwise operator) означава оператор, който действа над двоичното пред­ставяне на числовите типове. В компютрите всички данни и в част­ност числовите данни се представят като поредица от нули и единици. За целта се използва двоичната бройна система. Например числото 55 в двоична бройна система се представя като 00110111.

Действието на побитовите оператори над двоичните цифри 0 и 1 е показано в следната таблица:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **x** | **y** | **~x** | **x & y** | **x | y** | **x ^ y** |
| **1** | **1** | **0** | **1** | **1** | **0** |
| **1** | **0** | **0** | **0** | **1** | **1** |
| **0** | **1** | **1** | **0** | **1** | **1** |
| **0** | **0** | **1** | **0** | **0** | **0** |

Побитовите операции са важна част от една компютърна програма, защото те определят приоритета на различните действия, извършени от процесите, изпълняващи програмния код. Процесите имат функционалността да разделят програмата на разл