3. Adattípus, konstans, változó, kifejezés. Paraméterkiértékelés, paraméterátadás. Hatáskör, névterek, élettartam. Fordítási egységek, kivételkezelés.

Adattípus

A programok futásuk során a memóriában tárolják az adatokat. A nyers bájtsorozatok értelmezéséhez típusokat vezetnek be. Ez az adattípus meghatározza, hogy a tárolandó adat milyen szerkezetű, milyen műveletek hajthatóak végre rajta, milyen értékeket vehet fel, illetve hogyan kell tárolni a memóriában.

Az adattípusnak neve van, ami egy azonosító. Egy adattípust három dolog határoz meg, ezek:

– tartomány
– műveletek
– reprezentáció

Az adattípusok tartománya azokat az elemeket tartalmazza, amelyeket az adott típusú konkrét programozási eszköz fölvehet értékként. Az adattípushoz hozzátartoznak azok a műveletek, amelyeket a tartomány elemein végre tudunk hajtani. Minden adattípus mögött van egy megfelelő belső ábrázolási mód. A reprezentáció az egyes típusok tartományába tartozó értékek tárban való megjelenését határozza meg, tehát azt, hogy az egyes elemek hány bájtra és milyen bitkombinációra képződnek le.

Az adattípusoknak két nagy csoportjuk van:
Létezik egyszerű és összetett adattípus. Az egyszerűekhez tartoznak a skalárok (számok), a karakterek, sztringek, felsorolásos típus (enumerált) és a logikai. Az összetetthez tartozik a tömb (homogén) és a rekord vagy struktúra (heterogén). Programnyelvektől függően lehetnek speciális típusok is, például mutató, szótár, tuple.

Konstans

Olyan eszköz, mely az egész program futása során egy időben állandó értéket tárol, így már fordítási időben értéket kap. Lehet nevesített, amikor nevével tudunk rá hivatkozni a későbbiekben, és lehet név nélküli literál bármilyen kifejezésben.

A változó:

A változó olyan programozási eszköz, amelynek négy komponense van:
– név
– attribútumok
– cím
– érték

A ***név*** egy azonosító. A program szövegében a változó mindig a nevével jelenik meg, az viszont bármely komponenst jelentheti.

Az ***attribútumok*** olyan jellemzők, amelyek a változó futás közbeni viselkedését határozzák meg.

A változó ***címkomponense*** a tárnak azt a részét határozza meg, ahol a változó értéke elhelyezkedik. Ha futási időben nem változik, akkor statikus a tárolása, ha változik, akkor dinamikus
A változó ***értékkomponense*** mindig a címen elhelyezett bitkombinációként jelenik meg. A bitkombináció felépítését a típus által meghatározott reprezentáció dönti el.

Kifejezés

Műveletek és értékek sorozata, ami kiértékelődik, és egyetlen értékként jelenik meg a program futása közben. Ha konstans kifejezésről van szó (nincs benne változó), akkor már fordítási időben kiértékelődik, egyébként mindig futási időben kapnak értéket.
Egy kifejezés formálisan a következő összetevőkből áll:
Operandusok: az operandus literál, nevesített konstans, változó vagy függvényhívás lehet. Az értéket képviseli.
Operátorok: Műveleti jelek. Az értékekkel végrehajtandó műveleteket határozzák meg.
Kerek zárójelek: A műveletek végrehajtási sorrendjét befolyásolják. Minden nyelv megengedi a redundáns zárójelek alkalmazását.

Paraméterkiértékelés alatt értjük azt a folyamatot, amikor egy alprogram hívásánál egymáshoz rendelődnek a formális- és aktuális paraméterek, és meghatározódnak azok az információk, amelyek a paraméterátadásnál a kommunikációt szolgáltatják. A paraméterkiértékelésnél mindig a formális paraméter lista az elsődleges, ezt az alprogram specifikációja tartalmazza, egy darab van belőle. Aktuális paraméter lista viszont annyi lehet, ahányszor meghívjuk az alprogramot, ezeket rendeljük a formális paraméterlistához.
A formális és aktuális paraméterek egymáshoz rendelése történhet ***sorrendi kötés*** vagy ***névszerinti kötés*** szerint.
Sorrendi kötés esetén a formális paraméterekhez a felsorolás sorrendjében rendelődnek hozzá az aktuális paraméterek. Ezt minden nyelv ismeri, általában ez az alapértelmezés.
Névszerinti kötés esetén az aktuális paraméter listában határozhatjuk meg az egymáshoz rendelést, a formális paraméter nevét és mellette valamilyen szintaktikával az aktuális paramétert megadva. Ilyenkor lényegtelen a formális paraméterek sorrendje.

A paraméterátadás az alprogramok és más programegységek közötti kommunikáció egy formája. A paraméterátadásnál mindig van egy *hívó*, ez tetszőleges programegység és egy *hívott*, amelyik mindig alprogram. Kérdés, hogy melyik irányban és milyen információ mozog. A nyelvek *érték szerinti, cím szerinti, eredmény szerinti, érték-eredmény szerinti, név szerinti és szöveg szerinti* paraméterátadási módokat ismernek.

Érték szerinti: formális paraméterekbe másoljuk az aktuálisak értékeit.
Információáramlás egyirányú: hivó -> hívott

Cím szerinti: aktuálisak címét kapja meg az alprogram, így a hívó területen dolgozik.
Információáramlás: kétirányú.

Eredmény szerinti: aktuális cím átkerül, de saját lokális változókkal számol (értékmásolás), majd az eredményeket a megkapott címekre írja.
Információáramlás: egyirányú, hívott -> hívó

Érték-eredmény: érték és cím is átkerül, így nem kell a kapott címről kiolvasni az értékeket, rögtön érték szerinti átadás is történik.
Információáramlás: kétirányú, késztszer van értékmásolás

Név szerinti: átírja az alprogramban a formális paraméterek előfordulását a megadott aktuális paraméterek nevére, így azokon a változókon dolgozik, mintha az alprogram közvetlenül a kód hívott részére lenne írva.

Szöveg szerinti paraméterátadás a név szerintinek egy változata, annyiban különbözik tőle, hogy a hívás után az alprogram elkezd működni, az aktuális paraméter értelmező szövegkörnyezetének rögzítése, a formális paraméter csak akkor íródik felül, amikor a formális paraméter neve először fordul elő az alprogram szövegében a végrehajtás folyamán.

A hatáskör a nevekhez kapcsolódó fogalom. Egy név *hatásköre* alatt értjük a *program szövegének azon részét*, ahol az adott név ugyanazt a programozási eszközt hivatkozza, tehát jelentése, felhasználási módja, jellemzői azonosak.

Azt a tevékenységet, mikor egy név hatáskörét megállapítjuk, **hatáskörkezelésnek** hívjuk. Kétféle hatáskörkezelést ismerünk, a *statikus* és a *dinamikus* hatáskörkezelést.

Névtér

Osztályok, felsorolások, függvények, metódusok, és egyéb eszközök csoportosítását lehetővé tevő programnyelvi eszköz. Segítségükkel rövidebbé és olvashatóbbá tehetőek a kódok. Például C#-ban a 'using System.Console;' utasítás azt mondja, hogy használjuk a programban a „Systam.Console" névteret, így a 'System.Console.WriteLine();' helyett elegendő annyit írni, hogy 'WriteLine();'.

Fordítási egység

Olyan kódrészlet, mely külön is fordítható, és részben vagy teljesen független a kód többi részétől. A C# fordítási egysége a névtér, ami a C forrásállományának felel meg, és hatásköri egység is.

Élettartam

Egy elem élettartama a memóriába kerülésétől (definiálástól) a számára lefoglalt memóriaterület felszabadításáig tart, ami megtörténhet futási időben, vagy akkor, amikor a program futása véget ér.

Kivételkezelés

A kivételkezelési eszközrendszer azt teszi lehetővé, hogy az operációs rendszertől átvegyük a megszakítások kezelését, felhozzuk azt a program szintjére. A *kivételek* olyan események, amelyek megszakítást okoznak. A *kivételkezelés* az a tevékenység, amelyet a program végez, ha egy kivétel következik be. *Kivételkezelő* alatt egy olyan programrészt fogunk érteni, amely működésbe lép egy adott kivétel bekövetkezte után, reagálva az eseményre. A kivételkezelés az eseményvezérlés lehetőségét teszi lehetővé a programozásban. A kivételeknek általában van *neve* (egy kapcsolódó sztring, amely gyakran az eseményhez kapcsolódó üzenet szerepét játssza) és *kódja* (ami általában egy egész szám).