\Chapter{Célnyelvre fordítás}

Ezt már csak a közbülső reprezentáció és az adott nyelvek viszonyában kell megnézni.

Itt kellhet majd pretty printer féle. (Elegáns úgy megoldani, hogy opcióként választható legyen.)

Az alábbi részben a tervezett célnyelvek szintaxisának, elemeinek áttekintése történik.

\section{C}

A C szintaxisa az alábbiak szerint vizsgálható:

A C forráskódban minden utasítássor végét ;-vel kell jelölni, alapértelmezetten szekvenciálisan végrehajtódó kód, melyben a végrehajtás sorrendjének módosítására a szelekció, iteráció használható.

C nyelven az értékadás a változó típusmegjelölésével kezdődik majd a változó neve kell, hogy megadásra kerüljön, ezután egyenlőségjellel következik a változó értéke. A C nyelvben nincs láthatósági módosító, tehát ilyet nem kell megadni.

int valtozo = 12;

A C nyelvben a leggyakrabban használt típusok az alábbiak:

int – egyszerű szám típus

char – karakter típus

float – lebegőpontos

double – duplapontosságú lebegőpontos

void – a típus nélküli típust jelöli ez

array – tömb típus, általában []-ban adható meg, például char szoveg[] változóban tudunk szöveget tárolni

pointer változó – egy adott változó címére mutat, melyen keresztül a változó értéke módosítható, címszerinti átadás elvégezhető, például

int var = 10;

int \*alma = &var;

structure – struktúra típusú változó

A nyelvben összeszervezhetjük a többször lefuttatandó kódrészeket különféle függvényekbe, melyek esetében a visszatérési értéket kell megadni, majd a függvény nevét, végül zárójelben a paraméterlistát, ezután a függvénytörzsben ezen paramterek használhatók, módosíthatók, illetve új változók is létrehozhatók és használhatók. C nyelvben a függvényekből másik függvényt is meghívhatunk. A függvények megírásakor ügyelni kell rá, hogy a C a programkód sorrendjében fogja lefordítani és lefuttatni a programot, azaz, ha a kiemelt indító függvényben, a main függvényben olyan függvényhívást talál amit nem ismer, akkor fordításkor hibaüzenetet ad. Tehát a függvényeket vagy meg kell írni a main függvény előtt, vagy legalábbis egy függvény prototípust kell megadni. A függvény meghívásakor a paramétereket át lehet adni érték és cím szerint is, előbbi esetben a függvény lefutása után a paraméterül kapott változó értéke nem változik meg még akkor sem, ha azt a függvényen belül módosítottuk, utóbbi esetben viszont egy a változó címére mutató pointer kerül átadásra, melyen keresztül ténylegesen módosításra kerül a változó értéke.

Például egy függvény lehet az alábbi:

int osszeAd(int a, int b) {

return a+b;

}

Függvény prototípusnál csak a következőt kell megadni:

int osszeAd(int, int);

A C programnak van egy kiemelt függvénye, ez a main függvény, általában int visszatérési értékkel kell megadni, a legegyszerűbb programokban paramétere sincs, de megadható neki, hogy milyen paramétert várjon és azt a program indításakor szintén meg kell adni.

Minden C program egy main függvényt tartalmazhat és ez lesz a program belépési pontja, innen fog indulni a program.

A C nyelv ismeri a szelekció és iteráció vezérlési szerkezeteit. A szelekciót az if kulcsszóval, majd egy feltételvizsgálattal kell bevezetni, mely esetben az if utáni blokkban található kód vagy lefut, vagy nem annak függvényében, hogy a feltételvizsgálat milyen eredményt hoz. Megadható több ág is, az else if, kulcsszó segítségével, illetve ha egyik ágra sem tud belépni a program a feltételvizsgálatok miatt, megadható egy alapértelmezett else ág is. A szelekció másik formája a switch, mely szintén megtalálható a C nyelvben. Itt a switch után kell egy kifejezést megadni, melyet megpróbál ráilleszteni az egyes megadott elemekre, és az a kódrész fog lefutni ahol az illesztés sikeres volt. Ha sehol nem volt az, akkor megadható egy default kulcsszóval egy olyan kódrész mely mindenképpen lefut ha más kódrész nem tudott lefutni.

int a = 50;

if (a < 30) {

}

else if (a > 30 && a <50) {

}

else {

}

A fenti kódrészben sem az if, sem az else if ágra nem tud belépni a program, ezért az else ágban definiált utasítások fognak lefutni. If szelekció esetében több else if ág is megadható.

int a = 40;

switch(a) {

case 10:

break;

case 20:

break;

case 40:

break;

default:

}

A fenti kódban a switch elágazásra láthatunk egy példát, itt az első két ágra nem tud belépni a kód, de a harmadikra igen. Ilyen esetben a default ág felesleges is lehet, de általában előre nem tudjuk milyen kifejezést fogunk kapni. Az egyes blokkokban található break utasítás arra szolgál, hogy ha az adott ágban található kód lefutott, akkor lépjen ki a szelekcióból, más ágra már ne fusson rá.

A C nyelvben egy adott programkód többszöri egymás utáni ismétlése is megoldható az iteráció segítségével, az adott programkódot cilkusba szervezve.

A C nyelvben háromféle ilyen ciklusszervezési módszerrel találkozhatunk a while, a for és a do while. Előbbi kettő elöltesztelő, utóbbi hátultesztelő. Ez annyit jelent, az első két ciklus esetében azonnal megvizsgálja a program, hogy az adott feltételt elérte-e már a ciklus és ha igen akkor egyszer sem fog lefutni, a hátultesztelő ezzel szemben egyszer biztos lefut és csak utána tesztel.

A while ciklus:

int a = 10;

while (a > 0) {

a--;

}

A for ciklus:

for (int i = 10; i > 0; i--) {

}

és a do while ciklus:

int a = 10;

do {

a--;

} while(a > 0);

A C nyelv nem objektum orientált nyelv, azaz nincs kifejezetten osztály deklarációra lehetőség. A C nyelvben struktúrát lehet deklarálni, melynek lehetnek adattagjai, azaz benne definiált változók, de ezt nem definiálható benne függvény, csak függvénypointer adható meg benne.

struct StrukturaNev {

int a;

char b[];

} str;

str.a = 10;

\section{Java}

A Java nyelv egyike azon nyelveknek melyre nagy hatása volt a C nyelvnek, azonban a Java nyelv már a kezdetektől objektum orientált nyelvként került megalkotásra.

A Java kódban is az utasítássor végét ;-vel kell jelölni, valamint szintén szekvenciálisan végrehajtódó kód, melyben a végrehajtás sorrendjének módosítására a szelekció, iteráció használható.

Java nyelven is típusmegjelöléssel kezdődik a változó deklaráció majd a változó neve jön és végül a változó értéke következik.

A Java nyelvben már megjelenik a láthatósági módosít, tehát ezt is meg kell, illetve lehet adni. A Java nyelv 4 féle láthatósági módosítót különböztet meg, félnyilvános, protected, private és public. Mivel a Java nyelv osztályokra épül, alapértelmezetten is osztályokban írjuk a kódot. Ezeket az osztályokat csomagokba szervezzük, így a láthatósági módosítók az alábbiak alapján működnek:

public – az adott változót bárhonnan el lehet érni és módosítani

private – csak az adott osztályon belül lehet elérni a változót és módosítani is onnan lehet

protected – az adott változókat a csomagon belül, illetve a leszármazott osztályokból lehet elérni

félnyilvános – ilyenkor nem kell megadni semmilyen módosítót a változóhoz, a változó pedig a csomagon belül érhető el, tehát ez az alapértelmezett beállítás

Emellett olyan módosítók is léteznek a nyelvben mely nem a láthatóságot módosítják, hanem más funkciókat lehet velük beállítani:

static – az adott változókat osztályszintű változó lesz

final – az adott változók és metódusok, valamint osztályok melyek final metódussal vannak megjelölve, ezek értéke nem módosítható lesz

abstract – metódusok és osztályok kaphatják ezt, ezek absztrakt

synchronized – a szálkezeléskor alkalmazott módosítók

plédául

private int valtozo = 12;

public static final String STATIC\_VAR = „STRING”;

A C nyelvben a leggyakrabban használt típusok az alábbiak:

int – 32 bites szám típusú változó

char – karakter típus

float – lebegőpontos

double – duplapontosságú lebegőpontos

void – a típus nélküli típust jelöli ez

String – szöveg típusú változó

array – tömb típus, általában []-ban adható meg, például int[] numbers

A Java nyelvben is összeszervezhetjük a többször lefuttatandó kódrészeket különféle függvényekbe, melyek esetében a visszatérési értéket kell megadni, majd a függvény nevét, végül zárójelben a paraméterlistát, ezután a függvénytörzsben ezen paramterek használhatók, módosíthatók, illetve új változók is létrehozhatók és használhatók.

A C nyelvhez hasonlóan itt is a függvényekből másik függvényt is meghívhatunk. Ellentétben a feljebb tárgyalt nyelvvel, mivel a kód osztályokba van szervezve és ezen osztályok kerülnek példányosításra, ezért a függvényeket tetszőleges sorrendben megírhatók, ám az osztályban először a változókat kell megadni konvenció szerint.

C nyelvhez képest, Java nyelvben nincs pointer, így nincs cím szerinti átadás, érték szerint történnek ezek.

Egy függvény megadása Java nyelven is lehet az alábbi:

int osszeAd(int a, int b) {

return a+b;

}

A Java nyelvben is van egy kiemelt függvénye, a program belépési pontja. Ezt a függvényt is külön osztályban kell megírni, és ezt is main függvénynek adjuk meg.

public static void main(String[] args) {

}

A függvény mindenképpen osztály szintű metódus, tehát static módosítóval kell megadni és public a láthatósági köre. Emellett visszatérési értéke nincs, tehát void, valamint megadható paraméter indításkor, ezek az args nevű String tömbből lehet kiolvasni.

A Java nyelvben a C nyelvhez hasonló módon meg lehet adni szelekciót és iterációt, melyeket itt is az if, switch, illetve az iterációkat while, for és do while kulcsszavakkal lehet megadni, melyek után a programkód lefutása a C nyelven ismertekkel azonos. Az alább látható példa Java nyelvű szelekcióra.

private int a = 50;

if (a < 30) {

}

else if (a > 30 && a <50) {

}

else {

}

a = 40;

switch(a) {

case 10:

break;

case 20:

break;

case 40:

break;

default:

}

Itt is megtalálható szintén a break, mely utasítás arra szolgál, hogy ha az adott ágban található kód lefutott, akkor lépjen ki a szelekcióból.

Java nyelven is megtalálhatók a ciklusok melyek segítségével itt is adott kódrészek többször lefuttathatók. A ciklusok itt is a C nyelvhez hasonló működéssel rendelkeznek.

A while ciklus:

public int a = 10;

while (a > 0) {

a--;

}

A for ciklus:

private int i;

for (i = 10; i > 0; i--) {

}

és a do while ciklus:

int a = 10;

do {

a--;

} while(a > 0);

A Java nyelv objektum orientáltságának köszönhetően osztályokba szervezzük a kódot. Az osztályokat a class kulcsszóval kell bevezetni. Emellett az osztályoknál is meg lehet adni módosítókat. Osztályokat lehet interfészként is definiálni, ilyen osztályokban metódus prototípusok vannak definiálva. Ezeket az interfészeket be lehet implementálni az osztályokba.

Ezen felül az osztályok egymásból leszármaztathatók, a leszármazott osztályok öröklik a szülők adattagjait és metódusait. A Java nyelv az egyszeres öröklődést támogatja.

public class ElsoOsztaly {

int elso; int masodik; int harmadik = 0;

ElsoOsztaly(int elsoParam, int masodikParam) {

this.elso = elsoParam;

this.masodik = masodik Param;

}

int AddAll() {

return elso + masodik + harmadik;

}

}

public class Masodik extends ElsoOsztaly {

Masodik(int egy, int ketto) {

super(egy, ketto);

}

}

Masodik second = new Masodik(1, 3);

System.out.println(second.AddAll());

A fenti kódban az első osztályban lett megírva az összeadó függvény, majd az a második osztályon keresztül lett meghívva, és a második osztály lett leszármaztatva az első osztályból, erre az extends utasítást kell használni.

Emellett a fenti példában látható az osztály létrehozásának egyik fontos eleme, a konstruktor. A konstruktor egy kitüntetett függvény az osztályban, mely az osztály inicializálásakor mindig automatikusan lefut. Ebben lehet az elemeket inicializálni.

A Java nyelv megvalósítása alapján a konstruktor minden esetben lefut kivéve a private módosítóval ellátott konstruktor. Ha nincs megírva konstruktor akkor egy default konstruktor fut le. A konstruktor első utasítása minden esetben az ősosztály konstruktorának meghívása, akkor is ha nincs az kijelezve. Az ősosztály konstruktorának meghívása a super() metódussal történik. Ha nincs konkrétan megadott ősosztály, akkor az Object osztály konstruktora hívódik meg, mely minden osztály ősosztálya.

\section{PHP}

A PHP nyelv egy szerveroldali scriptnyelv, mely az egyik legelterjedtebb nyelv, legfrissebb változatai pedig már objektum orientáltan is lehet programozni.

A PHP kód minden esetben <?php és ?> tagek között kell megírni, .php fájlban kell a szerveroldalra feltölteni. A kódot html kódba lehet ágyazni, illetve fordítva is megtehető, hogy html kódba kerül beágyazásraa PHP kód.

PHP kódban szintén ;-vel kell jelölni az utasítások lezárását. Itt it megtalálható a szelekció és iteráció. A PHP gyengén típusos nyelv, azaz a változó deklarációkor nem kell megadni a változó típusát, csak a nevét és az értékét.

Láthatóság szerint public, private és protected módosítókat ismer a nyelv. A nyelv nem csomagokra épül, így a private adattagok csak az adott osztályban elérhetőek, a public mindenhol, a protected pedig az adott osztályban, valamint a leszármazottban és a szülőben érhető el.

A változókat a $ szimbólummal vezetjük be, ezután következik a változó neve, valamint az értéke:

$valtozo = 12;

Bár nem kell a típusokat kiírni a változókhoz, a háttérben természetesen van típusa az elemeknek. Ezek a típusok az alábbiak lehetnek:

Integer – egész szám típusú változó

Double – decimális számok

Boolean – kétértékű típus, lehet igaz vagy hamis

String – szöveg típus

NULL – egyedi típus, aminek csak egy értéke van a NULL

String – szöveg típusú változó

Array – tömb típus

Objects – osztályok példányának típusa

A PHP nyelvben is megtalálható a függvénydefiniálás lehetősége, mely a function kulcsszóval kerül bevezetésre, majd a függvény neve, végül zárójelben a paraméterlista kerül megadásra. Ezután a függvénytörzsben ezen paramterek használhatók, módosíthatók, illetve új változók is létrehozhatók és használhatók.

Az előző nyelvekhez hasonlóan itt is a függvényekből másik függvényt is meghívhatunk.

Általánosságban a PHP érték szerinti átadást végez, tehát az átadott paraméterek eredeti értékei nem változnak meg a függvényeken belüli módosítások hatására. Emellett a PHPban ismert a cím szerinti átadás is, mely esetben az átadott paraméter eredeti értéke is megváltozik a módosítás hatására. A cím szerinti átadást a C nyelvhez hasonlóan & jellel kell jelezni.

Egy függvény megadása PHP nyelven az alábbi lehet:

<?php

function novel($a) {

$a += 8;

}

function cimNovel(&$a) {

$a += 8;

}

$num = 10;

novel($num);

echo $num;

cimNovel($num);

echo $num;

?>

A fenti kódban az első függvény érték szerinti átadás után végez műveletet, a második függvény pedig cím szerint. Így a kód lefutása során az első kiírás 10 lesz, míg a második 18.

A PHP kódban nincs kiemelt függvény, a belépési pont az első kódsor, és utána szekvenciálisan fut le a kód.

Ezt a lefutást lehet befolyásolni a Java és C nyelvhez hasonló módon szelekcióval és iterációval, melyeket itt is az if, switch, illetve az iterációkat while, for és do while kulcsszavakkal lehet megadni, melyek után a programkód lefutása a C és Java nyelven ismertekkel azonos. Az alább látható példa Java nyelvű szelekcióra.

$a = 50;

if ($a < 30) {

}

else if ($a > 30 && $a <50) {

}

else {

}

$a = 40;

switch($a) {

case 10:

break;

case 20:

break;

case 40:

break;

default:

}

Itt is megtalálható szintén a break, mely utasítás arra szolgál, hogy ha az adott ágban található kód lefutott, akkor lépjen ki a szelekcióból.

A while ciklus:

$a = 10;

while ($a > 0) {

$a--;

}

A for ciklus:

for ($i = 10; $i > 0; $i--) {

}

és a do while ciklus:

$a = 10;

do {

$a--;

} while($a > 0);

A PHP nyelv megalkotásakor nem volt objektum orientált a nyelv, később azonban bevezették az osztálydefiníciót a nyelvbe. Az osztályokat a class kulcsszóval kell bevezetni, ezután kell megadni az osztály nevét.

Az osztály törzsében lehet adattagokat és metódusokat is definiálni. A PHPben a Java nyelvhez hasonlóan lehet megadni öröklődést is, mely esetben szintén a leszármazott osztályok öröklik a szülők adattagjait és metódusait.

<?php

class Elso {

var $elso;

function \_\_construct($el) {

$this->elso = $el;

}

function getElso() {

return $this->elso;

}

}

class Masodik extends Elso {

function \_\_construct($el) {

parent::\_\_construct($el);

}

}

$obj = new Masodik(5);

$obj->getElso();

?>

A fenti kódban az első osztályban lett megírva a lekérdező függvényt, de az a második osztályon keresztül lesz meghívva, mivel a második osztály lett leszármaztatva az első osztályból, erre az extends utasítást kell használni.

A fenti példában szintén megtalálható a konstruktor metódus is, mely a Java nyelvhez hasonlóan egy kitüntetett függvény, mely az osztály példányosításakor fut le és ebben lehet az elemeket inicializálni. Az ősosztály konstruktorát is meg lehet hívni a parent::\_\_construct() hívással.

\section{Python}

A Python nyelv egy egyszerűen és rugalmasan programozható objektum orientált nyelv.

Az adott nyelvben nem kell az utasítások sorát ;-vel lezárni, a sor vége jelzi az utasítás végét. Emellett a blokkokat jelző { és } is hiányzik, ezeket a tabulátorok jelzik.

A nyelv gyengén típusos, nem kell típust megadni a változók deklarálásakor.

elso = 55

A Python nyelvben is megtalálható a függvénydefiniálás lehetősége, mely a def kulcsszóval kerül bevezetésre, majd a függvény neve, végül zárójelben a paraméterlista kerül megadásra. Ezután : következik, majd új sorban tabulátorral kezdődik a függvénytörzs, ebben paramterek használhatók, módosíthatók, illetve új változók is létrehozhatók és használhatók.

def valtozas(parameter):

utasitas

A Python referencia szerinti átadást végez, tehát az eredeti változó is változni fog a metódusokon belüli módosítás hatására.

A program lefutását ezen nyelvben is lehet befolyásolni az előző nyelvekhez hasonló módon szelekcióval és iterációval, melyeket itt is az if, illetve az iterációkat while, for és kulcsszavakkal lehet megadni, amint az látható az alább példában.

var = 50

if var < 30:

elif var < 40:

else:

A while ciklus:

var = 0

while (var < 10):

var = var + 1

A for ciklus:

for num in range(10,20):

print num

Python nyelven definiálhatunk osztályokat is melyet az alábbi példa mutat be

class Elso:

count = 0

def \_\_init\_\_(self, name):

self.name = name

def displayCount(self):

print self.count

Az osztályban Pyhton nyelven is van kiemelt metódus, az \_\_init\_\_ névvel jelzett konstruktor, mely az adattagokat inicializálja. Látható, hogy mindegyik metódus a def kulcsszóval kezdődik és mindegyik paraméterlistájában megtalálható a self kulcsszó. Ezt a self-et a függvényhíváskor nem kell kiírni, ez automatikusan hozzáadásra kerül. Úgyszintén nem kell előre deklarálni a változókat melyeket a konstruktorban inicializálunk.

\section{JavaScript}

A JavaScript egy szkriptnyelv, mely leggyakrabban a böngészőkben használható, nyílt és platformfüggetlen nyelv.

Az adott nyelvben mindenképpen kell az utasítások sorának végére a ; az utasítások lezárása végett, valamint a Pythonnal ellentétben a { és } is kell mely a blokkok kezdetét és végét jelzi.

Emellett azonban a nyelv gyengén típusos, nem kell típust megadni a változók deklarálásakor, viszont kell egy var kulcsszó, mely a változókat bevezeti.

var elso = 55

Ebben a nyelvben a függvénydefiniálás szintén lehetséges, JavaScriptben a function kulcsszóval kell bevezetni. majd a függvény neve, végül zárójelben a paraméterlista kerül megadásra.

function valtozas(parameter) {

utasitas

}

A JavaScript alapértelmezetten érték szerinti átadást végez, tehát az eredeti változó ezen a módon nem fog módosulni. Ha szeretnénk a változó értékét mindenképpen módosítani akkor referenciaként kell átadni a változót, ahogy ezt az objektum orientált JavaScript részben látható.

A program lefutását ezen nyelvben is lehet befolyásolni az előző nyelvekhez hasonló módon szelekcióval és iterációval, melyeket itt is az if és switch illetve az iterációkat while, for és kulcsszavakkal lehet megadni, amint az látható az alább példában.

var = 50

if (var < 30) {

}

else if (var < 40) {

}

else {

}

A while ciklus:

var szam = 0

while (szam < 10) {

szam = szam + 1;

}

A for ciklus:

var count

for (count = 0; count < 10; count++){

document.write(count);

}

A JavaScript objektum orientált része olyan módon alakítható ki, hogy függvényeket írhatunk, melyben adattagokat és további függvényeket definiálhatunk.

class Elso(nev, mail, szoveg) {

this.nev = nev;

this.mail = mail;

this.szoveg = szoveg;

this.writeThis = function() {

console.log(this.nev + „” + this.mail);

}

}