Obligatorisk oppgave 2
-
Kort om oppgaven og litt informasjon

Fredrik Sørensen
OMS-gruppen, Ifi

Viktig

- Ny patch (patch_oblig2.zip) legges ut på kurssiden i dag.
- Oblig 1 vil bli rettet denne uken
  - Sjekk om det er registrert at den er levert (https://wwws.ifi.uio.no/)
- Frist var feil i oppgaveteksten, men riktig på oblig-siden
  - Fredag 9 mai
- Les listen med endringer og beskjeder!
- Det vil helt sikkert bli flere revisjoner av koden

Rettet i koden

- Eksemplet RunMe.d
- Referanser til compiler.NumberConversion
- Referanse til syntaxtree.BaseType
- Ikke teste new_type_not_struct_fail.d
- Ny kode i
  - bytecode.*
  - runtime.*
- OBS: Det er fortsatt feil i NumberConversion

Oppgaven

- Kan være nødvendig å endre på syntakstreet.
  - Abstrakte klasser som Expression, Statement, osv
- Sjekke om treet (programmet) oppfyller alle kravene i språknotatet
- Gi en kort feilmelding dersom en feil finnes.
- Kompilatoren skal svare med
  - (0) Godkjent
  - (1) Syntaksfeil
  - (2) Semantikkfeil
- Generere byte-kode ved hjelp av bibliotekklasser
- Teste semantikksjekken mot programmene i testsuiten
- Test bytecode-genereringen med eksemplet RunMe.d
- Rapporten teller!

Semantikksjekk

- Multiple deklarasjoner av ett navn på samme nivå må detekteres.
- Typekonvertering: int is-a float, int kan forfremmes til float i aritmetiske uttrykk, returverdier, og som aktuelt argumentuttrykk i funksjonskall.
- Main-funksjonen
  - Funksjonsdeklarasjonen til Main, må finnes på ytterste blokknivå av programmet.
  - Signaturen må matche func Main(), d.v.s. ingen argumenter og ingen returverdi.
- Funksjonsdeklarasjoners returverdi er:
  - ikke noe, eller
  - type i symboltabellen (predefinert eller brukerdefinert struct)

Semantikksjekk – Stmts

- AssignStmt
  - Variabelen på venstresiden må være deklarert.
  - Typene på vs. og hs. må være like, etter evt. typekonvertering.
- ReturnStmt
  - Forekommer bare i en funksjonsblokk (en FuncDecls liste av Stmts), ikke i de anonyme blokkene til IfStmts og WhileStmts.
  - Typen til uttrykket stemmer overens med funksjonens deklarerte type. Merk også særtilfelle uten returverdi, da blir argumenter til return-setningen en feil.
- WhileStmt: må ha en betingelse av typen bool.
- IfStmt: må ha en betingelse av typen bool.

Semantikksjekk – Exps

- NewExp: argumentet må være deklarert som en struct-type.
- Literals: Må være av en de forhåndsdefinerte typene i språket: float, int, string, bool eller null.
- Binære uttrykk (felles for alle operatorene bortsett fra negasjon) må ha samme typer, etter evt. typeforfremming, på begge sider av operatoren.
- Aritmetiske uttrykk: som for binære uttrykk, men typene må også begrenses til aritmetiske (int eller float).
- Logiske uttrykk (med "&&", "||" eller "!"): begge operander må være av typen bool.
- CallStmt
  - Navnet som kalles må være deklarert som funksjon.
  - Kallet må ha samme antall aktualparametre som formalparametre.
  - Aktualparametrene må ha samme type (eller mulig å tilordne) som formalparametrene.
  - Bruk av referanser ("ref") må stemme overens med funksjondeklarasjonen.
- Var
  - Evt. strukt-type må være deklarert.
  - Navnet må være deklarert.

Forskjeller i semantikken!

- Dere skal sjekke én semantikk for testprogrammene. Det er den ”offisielle” semantikken til språket D_b.
- Det er en begrenset semantikk for kodegenereringen.
- Det er altså **ikke** mulig/nødvendig å generere kode for alle eksemplene i katalogen *test*.

Forskjellene i semantikken er…

- Ikke blokkstruktur.
  - Den virtuelle maskinen har kun et globalt og lokalt nivå.
- Det er ikke referanseparamtere
  - basisparamterene overføre by-value
  - struktvariablene er pekerverdier og overføres for så vidt også by-value
  - Altså: Det er på sammen måte som i Java.
- Legg merke til at det kan ha blitt brukt litt andre navn, slik som at funksjonene kalles prosedyrer (Procedure).

Å bruke bytecode-biblioteket

```
CodeFile codeFile = new CodeFile();

// Her bygges bytekoden opp ...


byte[] bytecode = codeFile.getBytecode();

DataOutputStream stream =
  new DataOutputStream(
    new FileOutputStream("eks.bin"));

stream.write(bytecode); stream.close();
```

Å bruke bytecode-biblioteket

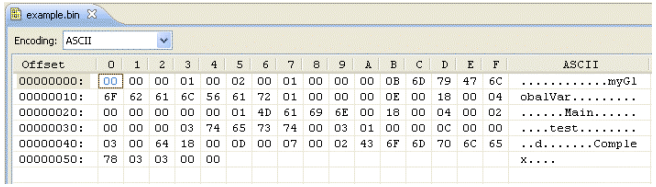
```
// Definerer først deklarasjonen
codeFile.addProcedure("Main");

// Lager metoden, variablen eller strukten
CodeProcedure main =
  new CodeProcedure("Main",
    VoidType.TYPE, codeFile);

// Legger til dens egenskaper
main.addInstruction(new RETURN());

// Oppdaterer Code-File-objektet
codeFile.updateProcedure(main);
```

Ant-targets

- clean
  - Bør kjøres før build hver gang.
- build
  - Bygger kompilatoren
- test
  - Kjører testene (ca. 40 stk)
- compile-runme
  - Kompilerer eksemplet RunMe.d
- list-runme
  - Lister ut innholdet i bytekoden laget for RunMe.d
- run-runme
  - Kjører RunMe.bin på den virtuelle maskinen

Sjekke/redigere binærfil

- Eclipse Hex Editor
  - http://ehep.sourceforge.net/update



Utfordringer

- Sjekke **alle** semantiske regler
- Klare å skille ut semantikken for kodegenerering
- Bruke biblioteket for kodegenerering
  - Holde orden på stack, med mer
  - Bruke klassene riktig

Tirsdag

- Flere detaljer om
  - Bytekoden
  - Den virtuelle maskinen
  - Tips til semantikksjekken
- Spørsmål
  - Send meg noe så snart det er noe dere lurer på

Husk fristen

Endelig frist **9. mai**

Ta kontakt

- Epost: fredrso@ifi.uio.no
- TLF: 22 85 04 73
- Kontor: 4311-NR (4. etasje)

SPØRSMÅL ???