Zadanie 1

Program w języku Instant składa się z ciągu instrukcji rozdzielonych średnikami.

Instrukcje są dwojakiego rodzaju:

- wyrażenie - wypisuje obliczoną wartość wyrażenia na standardowe wyjście
- przypisanie postaci `zmienna = wyrażenie` - przypisuje wartość wyrażenia na zmienną po lewej stronie; nic nie wypisuje

Wyrażenia składają się z literałów całkowitych nieujemnych, zmiennych i operacji arytmetycznych. Kolejność obliczenia argumentów operatora nie jest określona (można sobie wybrać wygodniejszą).

Składnia w formacie BNFC

```
Prog. Program ::= [Stmt] ;
      SAss. Stmt ::= Ident "=" Exp;
      SExp. Stmt ::= Exp ;
      separator Stmt ";" ;

      ExpAdd.           Exp1   ::= Exp2 "+"  Exp1 ;
      ExpSub.           Exp2   ::= Exp2 "-"  Exp3 ;
      ExpMul.           Exp3   ::= Exp3 "*"  Exp4 ;
      ExpDiv.           Exp3   ::= Exp3 "/"  Exp4 ;
      ExpLit.           Exp4   ::= Integer ;
      ExpVar.           Exp4   ::= Ident ;
      coercions Exp 4;
```

Uwaga:

- dodawanie wiąże **w prawo**
- przyjmujemy, że dodawanie i mnożenie są przemienne, ale nie są łączne.

Zadanie polega na napisaniu kompilatora dla języka Instant do JVM i LLVM.

W tym zadaniu wygenerowany kod powinien wykonywać wszystkie wyspecyfikowane operacje. Nie jest zatem na przykład dozwolone zastapienie wyrazenia 2+3 przez stałą 5, pominiecie przypisań na nieużywane zmienne itp. Usprawnianiem generowanego kodu zajmiemy się w kolejnych zadaniach.

Jedynym dozwolonym, a nawet pożądanym usprawnieniem jest wybór takiej kolejności obliczania podwyrażeń aby zminimalizować potrzebny rozmiar stosu JVM. W każdym wypadku potrzebny rozmiar stosu musi być obliczony i zadeklarowany (za podejścia typu "`limit stack 1000`" obcinamy punkty). Podobnie należy obliczyć i zadeklarować liczbę wykorzystywanych zmiennych lokalnych.

Wymagania techniczne

1. Projekt powinien być oddany w postaci spakowanego archiwum TAR (.tar.gz lub .tgz)

2. W korzeniu projektu (tj. w katalogu, w którym zostało rozpakowane archiwum) muszą się znajdować co najmniej:

    - Plik tekstowy README opisujący szczegóły kompilacji i uruchamiania programu, używane narzędzia i biblioteki, strukturę katalogów projektu, ewentualnie odnośniki do bardziej szczegółowej dokumentacji.
    - Plik Makefile (lub skrypt `build`) pozwalający na zbudowanie programu.
    - katalog src zawierający wyłącznie pliki źródłowe projektu (plus ewentualnie dostarczony przez nas plik Instant.cf); pliki pomocnicze takie jak biblioteki itp powinny być umieszczone w inych katalogach.
3. Program musi się kompilować na students poleceniem `make` (ewentualnie `./build`), uruchamianym w katalogu, w którym rozpakowano archiwum (czyli np. `tar xf foo.tgz; make`).

4. Wszelkie używane biblioteki (poza biblioteką standardową używanego jezyka programowania) muszą być opisane w README

5. Po zbudowaniu kompilatora, w korzeniu muszą się znajdować pliki wykonywalne o nazwie `insc_jvm` oraz `insc_llvm`

6. Wykonanie `insc_jvm foo/bar/baz.ins` dla poprawnego programu wejściowego `baz.ins` ma stworzyć pliki `baz.j` (kod Jasmin) oraz `baz.class` w katalogu `foo/bar` (przydatna może być opcja `-d` dla Jasmina). Wykorzystywany jasmin.jar należy umieścić w katalogu lib Ewentualne metody biblioteczne (`printInt` etc.) należy umieścić w klasie `Runtime.class` w katalogu lib

Wykonanie `insc_llvm foo/bar/baz.ins` dla poprawnego programu wejściowego `baz.ins` ma stworzyć pliki `baz.ll` (tekstowy kod LLVM) oraz `baz.bc` (bitkod LLVM wykonywalny przy uzyciu `lli`) w katalogu `foo/bar`

Punktacja:

Za to zadanie można uzyskać maksymalnie 6p. W przybliżeniu

- LLVM 2p
- JVM 3p
- Dla JVM: optymalizacja kolejności obliczania podwyrażeń, eliminacja zbędnych swap, wybór instrukcji: 1p.

Uwagi:

1. Kompilatory powinny działać w czasie nie gorszym niż O(n*log n) zwn rozmiar wejścia.
2. Dla JVM: będą odejmowane punkty za brak użycia specjalnych instrukcji dla małych stałych (iload_, icons_, bipush itp.).
3. Dla LLVM: można używać alloca, ale nie więcej niż jedno alloca na jedną zmienną.

Spóźnienia

Programy oddane po terminie będą obciążane karą 1p za każdy (rozpoczęty) tydzień opóźnienia. Ostatecznym terminem, po którym programy nie będą przyjmowane (“termin poprawkowy”) jest 3 grudnia.

Zasady

Projekt zaliczeniowy ma być pisany samodzielnie. Wszelkie przejawy niesamodzielności będą karane. W szczególności:

- nie wolno oglądać kodu innych studentów, pokazywać, ani w jakikolwiek sposób udostępniać swojego kodu
- wszelkie zapożyczenia powinny być opisane z podaniem źródła; dotyczy to także kodu wygenerowanego/zasugerowanego przez narzędzia AI i pokrewne (VS Code, Copilot, Claude, ChatGPT itp.).

Programy przykładowe

Paczka instant231015.tgz zawiera programy przykładowe i ich oczekiwane wyjście, oraz plik Instant.cf z gramatyką w formacie BNFC.