Univ rsité Paris 7 - L3 Informatiqu Anné 2006–2007

Introduction aux machines virtuelles

Examen du 4 mai 2007

Duré : 3h. Docum nts autorisés

I) Caml (∼6 points)

Exercice 1 Traduir n byt cod Caml l s phras s suivant s :

- `10*5/7`
- `let double x = x+x in double 10`
- `let = (1,2) in fst + snd`
  Indic : `fst` t `snd` sont traduit dir ct m nt n primitiv s sur l s blocs.
  Cit r un autr phras Caml qui pourrait donn r l mêm byt cod .

Exercice 2 Étant donné la portion d byt cod Caml suivant :

```
        closurerec 1, 0
        const [0: 1 [0: 2 0 ]]
        push
         cc 1
         pply 1
        push
        const 0
         ddint
        return 2
L1:      cc 0
        br nchifnot L2
         cc 0
        getfield 1
        push
        offsetclosure 0
         pply 1
        push
        const 1
         ddint
        return 1
L2:     const 0
        return 1
```

- On xécut c byt cod jusqu'au pr mi r passag par l'instruction `offsetclosure`. Décrir alors l'état d la machin virtu ll Caml après c tt instruction : cont nu d l'accumulat ur, d la pil t du tas. Justifi r brièv m nt.
- R trouv r un phras Caml dont la compilation produit c byt cod . Associ r à chaqu élém nt d c tt phras la parti byt cod qui corr spond. Qu l st l résultat d l' xécution ?

II) Ja a (∼6 points)

Exercice 3 Traduir n byt cod Java l s instructions suivant s (n sachant qu `x` t `y` sont l s variabl s local s 0 t 1).

- `y=3+2*(3+x));`
- `y=(x+3)*2+3;`
- `y=10; for (x=0; x*x<y; x++);`

Dans chacun d s cas, simul r l'évolution d la pil lors d l' xécution (on suppos qu `x` st nul initial m nt) t donn r la taill d pil néc ssair pour l bon déroul m nt du calcul.

Exercice 4 Étant donné la portion d byt cod Java suivant :

```
.method public st tic f([I)I
        .limit loc ls 3
        .limit st ck 3
        iconst_0
        istore_1
        iconst_0
        istore_2
L1:
        ilo d_2
         lo d_0
         rr ylength
        if_icmpge L2
        ilo d_1
         lo d_0
        ilo d_2
        i lo d
        i dd
        istore_1
        iinc 2 1
        goto L1
L2:
        ilo d_1
        ireturn
.end method
```

- Propos r un portion d cod Java dont la compilation produit c byt cod . Associ r à chaqu élém nt d cod Java la parti byt cod qui corr spond.
- Propos r un variant du byt cod ci-d ssus p rm ttant d minimis r l'usag d l'instruction `rr ylength`.

III) Logo (∼8 points)

On souhait cré r un machin virtu ll pour un mini-Logo. Pour mémoir , un Logo st un langag p rm ttant d réalis r d s d ssins n déplaçant un curs ur nommé "tortu ". C curs ur possèd n p rman nc un position xprimé par un pair d coordonné s ntièr s, t un ori ntation xprimé par un angl nti r n d gré.

a) Le langage Logo

Voici l s instructions d notr langag mini-Logo. Nous n l s utilis rons pas dir ct m nt, ll s n sont décrit s ici qu pour aid r à la compréh nsion d s instructions d byt cod d la s ction suivant .

- `Av nce n` : déplac la tortu d `n` dans sa dir ction courant , sans chang r son ori ntation.
- `Recule n` : déplac la tortu d `n` dans la dir ction opposé à son ori ntation actu ll . L'ori ntation st prés rvé lors du déplac m nt.
- `G uche n` : l'ori ntation st augm nté d `n` d grés.
- `Droite n` : l'ori ntation st diminué d `n` d grés.
- `LeveCr yon` : susp nd l tracé lors d s déplac m nts à v nir d la tortu .
- `B isseCr yon` : ncl nch l tracé lors d s déplac m nts à v nir d la tortu .
- `Repete n [ ... ]` : itèr `n` fois l s instructions prés nt s ntr l s croch ts.
- `Pour p ... Fin` : associ à la chaîn d caractèr s `p` la procédur constitué d la suit d'instructions allant jusqu'au `Fin`
- `p` : si la chaîn d caractèr s `p` corr spond à un procédur `Pour p ... Fin` , xécut l s instructions corr spondant s à c tt procédur

Par x mpl , un programm Logo d ssinant un carré p ut alors s'écrir :

```
Av nce 10 G uche 90 Av nce 10 G uche 90 Av nce 10 G uche 90 Av nce 10 G uche 90
```

ou aussi :

```
Repete 4 [ Av nce 10 G uche 90 ]
```

ou ncor :

```
Pour CARRE
 Repete 4 [ Av nce 10 G uche 90 ]
Fin
CARRE
```

b) Un bytecode Logo

Au li u d travaill r av c l s instructions "haut niv au" précéd nt s, on va utilis r un v rsion "byt cod ", dans laqu ll chaqu instruction corr spond à un nombr nti r ou d ux, l pr mi r nombr étant toujours l cod d l'instruction :

- `STOP` (cod =0) : arrêt l programm .
- `AVANCE n` (cod =1) : avanc la tortu d `n`, c tt quantité pouvant êtr négativ .
- `TOURNE n` (cod =2) : ajout `n` d grés à l'ori ntation d la tortu (`n` p ut êtr négatif).
- `LEVECRAYON` (cod =3)
- `BAISSECRAYON` (cod =4)
- `REPETE n` (cod =5) : instruction marquant l début d'un bloc à répét r `n` fois
- `FINREPETE o` (cod =6) : fin d'un zon à répét r. L' nti r `o` st un décalag (offs t) p r-m ttant d r v nir just après l'instruction `REPETE` corr spondant (cf. x mpl ci-d ssous).
- `FINPOUR` (cod =7) : instruction marquant la fin d'un procédur t l r tour à l' ndroit d l'app l.

- `LANCE o` (cod =8) : lanc m nt d la procédur situé o nti rs plus loin dans l byt cod .

Ainsi l s trois x mpl s précéd nts corr spond nt aux trois portions d byt cod suivants :

```
1 10 2 90 1 10 2 90 1 10 2 90 1 10 0
```

```
5 4 1 10 2 90 6 -6 0
```

```
8 1 0 5 4 1 10 2 90 6 -6 7
```

Exercice 5

1. On considèr d'abord un Machin Virtu ll Logo (MVL) n traitant qu l s instructions d byt cod 0 à 4. Qu ls sont l s r gistr s néc ssair s au fonctionn m nt d c tt MVL ? L'usag d'un pil st-il néc ssair ?
2. Qu faut-il n plus pour obt nir un MVL traitant aussi l s instructions 5 t 6 ? Votr solution p rm t- ll d'imbriqu r d s répétitions ? Comm nt trait r c cas ?
3. Qu faut-il n plus pour obt nir un MVL traitant aussi l s instructions 7 t 8 ? Votr solution p rm t- ll d'imbriqu r d s app ls d procédur s dans d'autr s procédur s ? Comm nt trait r c cas ?
4. L'argum nt `o` d l'instruction 6 (`FINREPETE`) st-il indisp nsabl ? Si c n' st pas l cas, qu faut-il n plus comm structur pour s' n pass r ?

Exercice 6 En utilisant l langag d votr choix (parmi Ocaml, Java t C), écrir un Machin Virtu ll Logo, c' st-à-dir un fonction r c vant un tabl au d nombr s corr spondant à du byt cod Logo t réalisant l d ssin corr spondant.

On pourra suppos r l' xist nc d'un fonction `dessine_ligne` qui att nd quatr nti rs x_1 y_1 x_2 y_2 t trac l s gm nt ntr (x_1, y_1) t (x_2, y_2).

Indication : Dans un pr mi r t mps, vous pouv z vous conc ntr r sur la réalisation d'un machin virtu ll traitant s ul m nt l s instructions 0 à 4, puis 0 à 6. Évid mm nt, à vaincr sans péril, on triomph sans gloir , t on a moins d points...