ILP - Examen réparti novembre 2013

C. Queinnec

Introduction

Lisez bien les exemples.

```
{ m0 = memoryGet();                    // 1 allocation fa
  1 + 2;                               // 3 allocations
  print(memoryGet() - m0); // 64 // 1 allocation fa
  memoryReset();
  print(memoryGet());          //  0 // 1 allocation fa
  print(memoryGet());          // 16 // 1 allocation fa
}
```

Sur cette machine, un entier ILP occupe 16 octets.

Question 1a

On demandait de l'ILP pas du C !
La fonction memoryGet alloue nécessairement un entier donc

```
function alloue (n) {
   while ( memoryGet() < n ) {
   }
}
```

Question 1b

```
// Test de alloue
{ m0 = memoryGet()
  n = 100
  alloue(n)
  n < memoryGet()
}
// < pas = !
```

```
// test de memoryReset
{ n = 100
  alloue(n)
  memoryReset()
  m1 = memoryGet()
  m1 == 0
 }
```

Question 2

```
long ILP_bytes_count = 0;

ILP_Object ILP_malloc (int size, enum ILP_Kind kind) {
  ILP_Object result = malloc(size);
  if ( result == NULL ) {
    return ILP_die("Memory exhaustion");
  };
  ILP_bytes_count += size;
  result->_kind = kind;
  return result;
}

ILP_Object ILP_memoryGet () {
  return ILP_Integer2ILP(ILP_bytes_count);
}

ILP_Object ILP_memoryReset () {
  ILP_bytes_count = 0;
  return ILP_FALSE;
}
```

Placer ILP_malloc avant libilp.a. Ramener des valeurs ILP pas des int ou long

Question 3a

Transformation de programmes (quelle que soit la constante sauf Booléens) :

```
... constante ...                    globalN = constante
                          =>         ... globalN ...
```

En assurant qu'une même constante n'engendre qu'une seule variable globale.

Question 4

```
struct ILP_Object ILP_object_42 = {
  ILP_INTEGER_KIND, { 42 } };
#define ILP_42 (&ILP_object_42)

struct ILP_Object ILP_object_43 = {
  ILP_INTEGER_KIND, { 43 } };
#define ILP_43 (&ILP_object_43)

#define ILP_Integer2ILP(i) ILP_make_integer_new(i)
/* Warning */
ILP_Object ILP_make_integer_new (int d) {
  switch (d) {
    case 42: return ILP_42;
    case 43: return ILP_43;
    default: return ILP_make_integer(d);
  }
}
```