Programming Languages I:G221.2110
Solution to HW 3

B. Mishra

October 23, 1995

Disclaimer *Beware, the programs shown here have not been compiled, debugged, and run. So there might be errors. Nonetheless, you should get the general idea of how the solutions go.*

4.2 & 4.3 (pp. 162, Sethi).

Problem 2.

```
generic
  type ITEM is private;
  type SORT_ARRAY is array(POSITIVE range<>) of ITEM;
  with function ``<´´(U,V: ITEM) return BOOLEAN is <>;

function HEAP_SORT(A: in SORT_ARRAY) return SORT_ARRAY;

function HEAP_SORT(A: in SORT_ARRAY) return SORT_ARRAY is
  ARRAY_SIZE, HEAP_SIZE: POSITIVE := A´RANGE;
  FIRST: integer := A´FIRST;
  LAST: integer := A´LAST;
  B: SORT_ARRAY(FIRST..LAST) := A;
  I: POSITIVE range 0..ARRAY_SIZE;

  procedure SWAP(I, J: in POSITIVE) is
    temp: ITEM := B(I);
  begin
    B(I) := B(J);
    B(J) := temp;
    return;
  end SWAP;

  procedure HEAPIFY(I: in POSITIVE; HEAP_SIZE: in POSITIVE) is
    L, R, SMALLEST: POSITIVE range 0..ARRAY_SIZE;
  begin
    L := I*2;
    R := I*2 + 1;
    if (L <= HEAP_SIZE) and (B(L) < B(I)) then
      SMALLEST := L;
    else
      SMALLEST := I;
    end if;
    if (R <= HEAP_SIZE) and (B(R) < B(SMALLEST)) then
      SMALLEST := R;
    end if;
    if (SMALLEST /= I) then
      SWAP(I, SMALLEST);
```

```
    end if;
    return;
  end HEAPIFY;

begin
  for I in reverse FIRST..((ARRAY_SIZE/2)+FIRST) loop
    HEAPIFY(I, HEAP_SIZE);
  end loop;

  for I in reverse (FIRST-1)..LAST loop
    SWAP(FIRST, I);
    HEAP_SIZE := HEAP_SIZE - 1;
    HEAPIFY(FIRST, HEAP_SIZE);
  end loop;

  return B;
end HEAP_SORT;
```

Problem 3.

- PASCAL

```
type item = record
  coeff: integer;
  power: integer;
end;

type poly = array[1..10] of item;
var ipoly: poly;
var num: integer;

procedure PolyDiff;
  var i: integer;
begin
  for i := 1 to num do
  begin
    ipoly[i].coeff := ipoly[i].coeff * ipoly[i].power;
    ipoly[i].power := ipoly[i].power - 1;
  end;
end;
```

- Ada

```
with text_io;
use text_io;
package poly is
  --...
  procedure PolyDiff;
end poly;

package body Poly is
  --...

  type item is record
    coeff: integer;
    power: positive
  end record;
```

```
  type poly is array(1..10) of item;
    ipoly: poly;
    num: integer;
    i, iterate: integer;

  procedure PolyDiff is
    i: integer;
  begin
    for i in 1..num loop
    begin
      ipoly(i).coeff := ipoly(i).coeff * ipoly(i).power;
      ipoly(i).power := ipoly(i).power - 1;
    end;
  end;
end poly;
```

- C++

```
typedef struct item{
  int coeff;
  int power;
}

class poly{
  public:
    //...
    void PolyDiff(void);
  private:
    struct item ipoly[10]
    int num;
}

void poly::PolyDiff(void){
  int i;
  for(i=1; i <= num; i++){
    ipoly[i].coeff *= ipoly[i].power;
    (ipoly[i].power)--;
  }
}
```