// t7 Decaf Classes with inheritance

class Animal {

 int height;

 Animal mother;

 void InitAnimal(int h, Animal mom) {

 this.height = h;

 mother = mom;

 }

 int GetHeight() {

 return height;

 }

 Animal GetMom() {

 return this.mother;

 }

}

class Cow extends Animal {

 bool isSpotted;

 void InitCow(int h, Animal m, bool spot) {

 isSpotted = spot;

 InitAnimal(h,m);

 }

 bool IsSpottedCow () {

 return isSpotted;

 }

}

void main() {

 Cow betsy;

 Animal b;

 betsy = New(Cow);

 betsy.InitCow(5, null, true);

 b = betsy;

 b.GetMom();

 Print("spots: ",betsy.IsSpottedCow(), " height: ", b.GetHeight());

}

**Sample Generated TAC:**

\_Animal.InitAnimal:

 BeginFunc 0 ;

 \*(this + 4) = h ;

 \*(this + 8) = mom ;

 EndFunc ;

\_Animal.GetHeight:

 BeginFunc 4 ;

 \_tmp0 = \*(this + 4) ;

 Return \_tmp0 ;

 EndFunc ;

\_Animal.GetMom:

 BeginFunc 4 ;

 \_tmp1 = \*(this + 8) ;

 Return \_tmp1 ;

 EndFunc ;

VTable Animal =

 \_Animal.InitAnimal,

 \_Animal.GetHeight,

 \_Animal.GetMom,

;

\_Cow.InitCow:

 BeginFunc 8 ;

 \*(this + 12) = spot ;

 \_tmp2 = \*(this) ;

 \_tmp3 = \*(\_tmp2) ;

 PushParam m ;

 PushParam h ;

 PushParam this ;

 ACall \_tmp3 ;

 PopParams 12 ;

 EndFunc ;

\_Cow.IsSpottedCow:

 BeginFunc 4 ;

 \_tmp4 = \*(this + 12) ;

 Return \_tmp4 ;

 EndFunc ;

VTable Cow =

 \_Animal.InitAnimal,

 \_Animal.GetHeight,

 \_Animal.GetMom,

 \_Cow.InitCow,

 \_Cow.IsSpottedCow,

;

main:

 BeginFunc 84 ;

 \_tmp5 = 16 ;

 PushParam \_tmp5 ;

 \_tmp6 = LCall \_Alloc ;

 PopParams 4 ;

 \_tmp7 = Cow ;

 \*(\_tmp6) = \_tmp7 ;

 betsy = \_tmp6 ;

 \_tmp8 = 5 ;

 \_tmp9 = 0 ;

 \_tmp10 = 1 ;

 \_tmp11 = \*(betsy) ;

 \_tmp12 = \*(\_tmp11 + 12) ;

 PushParam \_tmp10 ;

 PushParam \_tmp9 ;

 PushParam \_tmp8 ;

 PushParam betsy ;

 ACall \_tmp12 ;

 PopParams 16 ;

 b = betsy ;

 \_tmp13 = \*(b) ;

 \_tmp14 = \*(\_tmp13 + 8) ;

 PushParam b ;

 \_tmp15 = ACall \_tmp14 ;

 PopParams 4 ;

 \_tmp16 = "spots: " ;

 PushParam \_tmp16 ;

 LCall \_PrintString ;

 PopParams 4 ;

 \_tmp17 = \*(betsy) ;

 \_tmp18 = \*(\_tmp17 + 16) ;

 PushParam betsy ;

 \_tmp19 = ACall \_tmp18 ;

 PopParams 4 ;

 PushParam \_tmp19 ;

 LCall \_PrintBool ;

 PopParams 4 ;

 \_tmp20 = " height: " ;

 PushParam \_tmp20 ;

 LCall \_PrintString ;

 PopParams 4 ;

 \_tmp21 = \*(b) ;

 \_tmp22 = \*(\_tmp21 + 4) ;

 PushParam b ;

 \_tmp23 = ACall \_tmp22 ;

 PopParams 4 ;

 PushParam \_tmp23 ;

 LCall \_PrintInt ;

 PopParams 4 ;

 EndFunc ;

// t8 Decaf Classes with overriding

class Squash extends Vegetable {

 void Grow(Seeds []seeds, int [][]water)

 {

 Print("But I don't like squash\n");

 Print(10 \* 5);

 }

}

class Vegetable {

 int weight;

 int color;

 void Eat(Vegetable veg)

 {

 Seeds[] s;

 int [][]w;

 color = 5 % 2;

 Print("Yum! ", color, "\n");

 veg.Grow(s, w);

 return;

 }

 void Grow(Seeds []seeds, int [][]water)

 {

 Print("Grow, little vegetables, grow!\n");

 Eat(this);

 }

}

void Grow(int a) {

 Print("mmm... veggies!\n");

}

class Seeds {

 int size;

}

void main()

{

 Vegetable []veggies;

 veggies = NewArray(2, Vegetable);

 veggies[0] = New(Squash);

 veggies[1] = New(Vegetable);

 Grow(10);

 veggies[1].Eat(veggies[0]);

}

**Generated TAC:**

\_Squash.Grow:

 BeginFunc 16 ;

 \_tmp0 = "But I don't like squash\n" ;

 PushParam \_tmp0 ;

 LCall \_PrintString ;

 PopParams 4 ;

 \_tmp1 = 10 ;

 \_tmp2 = 5 ;

 \_tmp3 = \_tmp1 \* \_tmp2 ;

 PushParam \_tmp3 ;

 LCall \_PrintInt ;

 PopParams 4 ;

 EndFunc ;

VTable Squash =

 \_Vegetable.Eat,

 \_Squash.Grow,

;

\_Vegetable.Eat:

 BeginFunc 40 ;

 \_tmp4 = 5 ;

 \_tmp5 = 2 ;

 \_tmp6 = \_tmp4 % \_tmp5 ;

 \*(this + 4) = \_tmp6 ;

 \_tmp7 = "Yum! " ;

 PushParam \_tmp7 ;

 LCall \_PrintString ;

 PopParams 4 ;

 \_tmp8 = \*(this + 4) ;

 PushParam \_tmp8 ;

 LCall \_PrintInt ;

 PopParams 4 ;

 \_tmp9 = "\n" ;

 PushParam \_tmp9 ;

 LCall \_PrintString ;

 PopParams 4 ;

 \_tmp10 = \*(veg) ;

 \_tmp11 = \*(\_tmp10 + 4) ;

 PushParam w ;

 PushParam s ;

 PushParam veg ;

 ACall \_tmp11 ;

 PopParams 12 ;

 Return ;

 EndFunc ;

\_Vegetable.Grow:

 BeginFunc 12 ;

 \_tmp12 = "Grow, little vegetables, grow!\n" ;

 PushParam \_tmp12 ;

 LCall \_PrintString ;

 PopParams 4 ;

 \_tmp13 = \*(this) ;

 \_tmp14 = \*(\_tmp13) ;

 PushParam this ;

 ACall \_tmp14 ;

 PopParams 8 ;

 EndFunc ;

VTable Vegetable =

 \_Vegetable.Eat,

 \_Vegetable.Grow,

;

\_Grow:

 BeginFunc 4 ;

 \_tmp15 = "mmm... veggies!\n" ;

 PushParam \_tmp15 ;

 LCall \_PrintString ;

 PopParams 4 ;

 EndFunc ;

VTable Seeds =

;

main:

 BeginFunc 264 ;

 \_tmp16 = 2 ;

 \_tmp17 = 0 ;

 \_tmp18 = \_tmp16 < \_tmp17 ;

 IfZ \_tmp18 Goto \_L0 ;

 \_tmp19 = "Decaf runtime error: Array size is <= 0\n" ;

 PushParam \_tmp19 ;

 LCall \_PrintString ;

 PopParams 4 ;

 LCall \_Halt ;

\_L0:

 \_tmp20 = 1 ;

 \_tmp21 = \_tmp20 + \_tmp16 ;

 \_tmp22 = 4 ;

 \_tmp23 = \_tmp21 \* \_tmp22 ;

 PushParam \_tmp23 ;

 \_tmp24 = LCall \_Alloc ;

 PopParams 4 ;

 \*(\_tmp24) = \_tmp16 ;

 \_tmp25 = \_tmp24 + \_tmp22 ;

 veggies = \_tmp25 ;

 \_tmp26 = 0 ;

 \_tmp27 = 0 ;

 \_tmp28 = \_tmp26 < \_tmp27 ;

 \_tmp29 = \*(veggies + -4) ;

 \_tmp30 = \_tmp26 < \_tmp29 ;

 \_tmp31 = \_tmp30 == \_tmp27 ;

 \_tmp32 = \_tmp28 || \_tmp31 ;

 IfZ \_tmp32 Goto \_L1 ;

 \_tmp33 = "Decaf runtime error: Array subscript out of bound..." ;

 PushParam \_tmp33 ;

 LCall \_PrintString ;

 PopParams 4 ;

 LCall \_Halt ;

\_L1:

 \_tmp34 = 4 ;

 \_tmp35 = \_tmp34 \* \_tmp26 ;

 \_tmp36 = veggies + \_tmp35 ;

 \_tmp37 = 12 ;

 PushParam \_tmp37 ;

 \_tmp38 = LCall \_Alloc ;

 PopParams 4 ;

 \_tmp39 = Squash ;

 \*(\_tmp38) = \_tmp39 ;

 \*(\_tmp36) = \_tmp38 ;

 \_tmp40 = 1 ;

 \_tmp41 = 0 ;

 \_tmp42 = \_tmp40 < \_tmp41 ;

 \_tmp43 = \*(veggies + -4) ;

 \_tmp44 = \_tmp40 < \_tmp43 ;

 \_tmp45 = \_tmp44 == \_tmp41 ;

 \_tmp46 = \_tmp42 || \_tmp45 ;

 IfZ \_tmp46 Goto \_L2 ;

 \_tmp47 = "Decaf runtime error: Array subscript out of bound..." ;

 PushParam \_tmp47 ;

 LCall \_PrintString ;

 PopParams 4 ;

 LCall \_Halt ;

\_L2:

 \_tmp48 = 4 ;

 \_tmp49 = \_tmp48 \* \_tmp40 ;

 \_tmp50 = veggies + \_tmp49 ;

 \_tmp51 = 12 ;

 PushParam \_tmp51 ;

 \_tmp52 = LCall \_Alloc ;

 PopParams 4 ;

 \_tmp53 = Vegetable ;

 \*(\_tmp52) = \_tmp53 ;

 \*(\_tmp50) = \_tmp52 ;

 \_tmp54 = 10 ;

 PushParam \_tmp54 ;

 LCall \_Grow ;

 PopParams 4 ;

 \_tmp55 = 0 ;

 \_tmp56 = 0 ;

 \_tmp57 = \_tmp55 < \_tmp56 ;

 \_tmp58 = \*(veggies + -4) ;

 \_tmp59 = \_tmp55 < \_tmp58 ;

 \_tmp60 = \_tmp59 == \_tmp56 ;

 \_tmp61 = \_tmp57 || \_tmp60 ;

 IfZ \_tmp61 Goto \_L3 ;

 \_tmp62 = "Decaf runtime error: Array subscript out of bound..." ;

 PushParam \_tmp62 ;

 LCall \_PrintString ;

 PopParams 4 ;

 LCall \_Halt ;

\_L3:

 \_tmp63 = 4 ;

 \_tmp64 = \_tmp63 \* \_tmp55 ;

 \_tmp65 = veggies + \_tmp64 ;

 \_tmp66 = \*(\_tmp65) ;

 \_tmp67 = 1 ;

 \_tmp68 = 0 ;

 \_tmp69 = \_tmp67 < \_tmp68 ;

 \_tmp70 = \*(veggies + -4) ;

 \_tmp71 = \_tmp67 < \_tmp70 ;

 \_tmp72 = \_tmp71 == \_tmp68 ;

 \_tmp73 = \_tmp69 || \_tmp72 ;

 IfZ \_tmp73 Goto \_L4 ;

 \_tmp74 = "Decaf runtime error: Array subscript out of bound..." ;

 PushParam \_tmp74 ;

 LCall \_PrintString ;

 PopParams 4 ;

 LCall \_Halt ;

\_L4:

 \_tmp75 = 4 ;

 \_tmp76 = \_tmp75 \* \_tmp67 ;

 \_tmp77 = veggies + \_tmp76 ;

 \_tmp78 = \*(\_tmp77) ;

 \_tmp79 = \*(\_tmp78) ;

 \_tmp80 = \*(\_tmp79) ;

 PushParam \_tmp66 ;

 PushParam \_tmp78 ;

 ACall \_tmp80 ;

 PopParams 8 ;

 EndFunc ;