ECE326 – Fall 2019: Week 4 Exercise Questions

1. True or False **[1 mark each]**

Circle T is true, otherwise circle F for false.

1. With C3 Linearization, Python completely solves the diamond problem. [✓ **T**] **F**

2. 0x8888FEDC is a 4-byte aligned address. [✓ **T**] **F**

3. Suppose class A is inherited by class B, and C, monotonicity guarantees that A will behave the same for both B and C. [✓ **T**] **F**

4. Adding a new pure virtual function to a base class with many existing derived classes is an example of a fragile base class problem. **T** [✓ **F**]

5. The main difference between delegation and type embedding is that with type embedding, you can no longer reference the embedded member by name. [✓ **T**] **F**

2. Multiple Answers **[2 marks each]**

Pick all answer(s) that are correct. You will lose 1 mark per wrong choice, down to 0 marks.

1. Which of the following are true about mixins?

   (a) It requires subclass to complete its implementation. Not Necessarily.
   [✓ (b)] It can contain both member variables and functions.
   [✓ (c)] It is used as a super type to the derived class.
   (d) Using it requires method forwarding. Only composition requires forwarding.
   [✓ (e)] The order in which mixins are composed may change behaviour of the subclass.

2. Java only supports single inheritance with runtime polymorphism. Which of the following is true?

   [✓ (a)] Java does not support mixins. Mixin requires multiple inheritance.
   (b) Java does not need virtual tables. Still required to implement dynamic dispatch.
   [✓ (c)] Casting pointers (internally, Java does not expose pointers to programmers) in Java will never require point offsetting.
   [✓ (d)] Java does not need to deal with inheritance-related ambiguity.
   [✓ (e)] Java does not have method resolution order. Only late binding languages require MRO.

3. Virtual Base Class in C++ **[10 marks]**

Draw the data layout of class X (include padding assuming 8-byte alignment, and write down the size of each sub-structure) and all the virtual tables generated for class X and its ancestors.

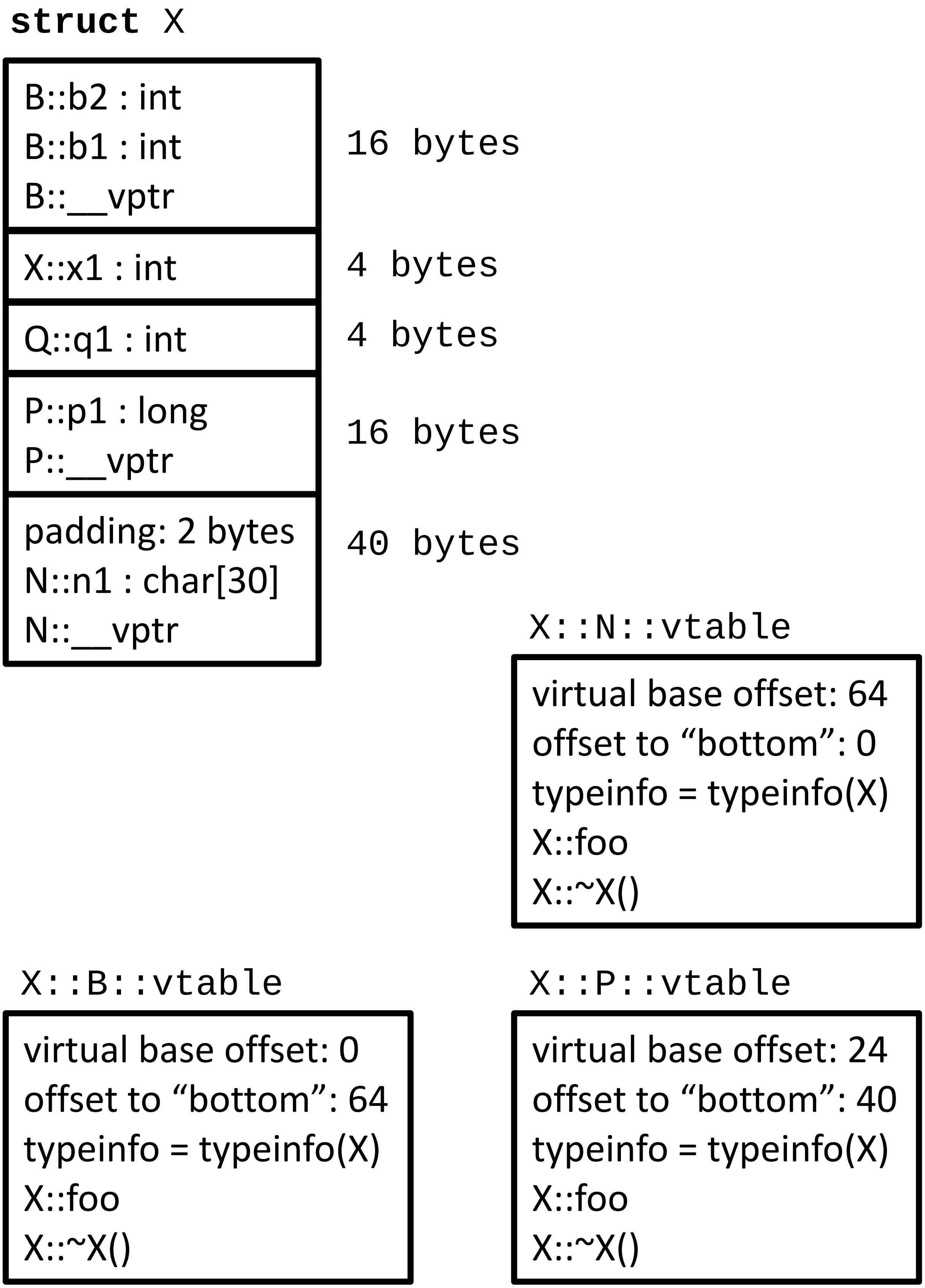
```
struct B {
  int b1;
  int b2;
  virtual void foo() { cout << “A.foo”; }
  virtual ~A() {}
};

struct P : virtual public B {
  long p1;
  virtual void foo() override { cout << “P.foo”; }
};

struct Q : public P {
  int q1;
};

struct N : virtual public B {
  char n1[30];
};

struct X : public N, public Q {
  int x1;
  virtual void foo() override { cout << “X.foo”; }
};
```



struct X

B::b2 : int B::b1 : int B::__vptr	16 bytes
X::x1 : int	4 bytes
Q::q1 : int	4 bytes
P::p1 : long P::__vptr	16 bytes
padding: 2 bytes N::n1 : char[30] N::__vptr	40 bytes

X::N::vtable

virtual base offset: 64
offset to “bottom”: 0
typeinfo = typeinfo(X)
X::foo
X::~X()

X::B::vtable

virtual base offset: 0
offset to “bottom”: 64
typeinfo = typeinfo(X)
X::foo
X::~X()

X::P::vtable

virtual base offset: 24
offset to “bottom”: 40
typeinfo = typeinfo(X)
X::foo
X::~X()

4. Method Resolution Order **[10 marks]**

a. For the following inheritance hierarchy in Python, draw a diagram of the hierarchy. [2 marks]

```
class A: pass
class B: pass
class C: pass
class D: pass
class E: pass
class P(A, B, C): pass
class Q(D, B, E): pass
class R(D, A): pass
class X(P, R, Q): pass
```

b. What is the C3 Linearization of X? [8 marks]

L[A] = (A, o)

L[B] = (B, o)

L[C] = (C, o)

L[D] = (D, o)

L[E] = (E, o)

L[P] = (P, A, B, C, o)

L[Q] = (Q, D, B, E, o)

L[R] = (R, D, A, o)

L[X]

= (X, merge((P, A, B, C, o), (R, D, A, o), (Q, D, B, E, o), (P, R, Q)))

= (X, P, merge((A, B, C, o), (R, D, A, o), (Q, D, B, E, o), (R, Q)))

= (X, P, R, merge((A, B, C, o), (D, A, o), (Q, D, B, E, o), (Q)))

= (X, P, R, Q, merge((A, B, C, o), (D, A, o), (D, B, E, o)))   # A bad head, in tail of 2nd list

= (X, P, R, Q, D, merge((A, B, C, o), (A, o), (B, E, o)))

= (X, P, R, Q, D, A, merge((B, C, o), (o), (B, E, o)))

= (X, P, R, Q, D, A, B, merge((C, o), (o), (E, o)))

= (X, P, R, Q, D, A, B, C, E, o)