CS 251
Intermediate Programming
Quiz 7

Brooke Chenoweth

University of New Mexico

Fall 2026

Question 1

Which of the following does *not* correctly declare and instantiate a map that associates `String` keys with `Double` values?

A

```
Map<String, Double> map = new HashMap<String, Double>();
```

B

```
HashMap<String, Double> map = new HashMap<String, Double>();
```

C

```
Map<String, Double> map = new TreeMap<String, Double>();
```

D `Map<String, Double> map = new HashMap<>();`

E `Map<String, Double> map = new TreeMap<>();`

F `Map<String, Double> map = new Map<>();`

G All of the above.

H None of the above.

Question 1

Which of the following does *not* correctly declare and instantiate a map that associates `String` keys with `Double` values?

A
```
Map<String, Double> map = new HashMap<String, Double>();
```
B
```
HashMap<String, Double> map = new HashMap<String, Double>();
```
C
```
Map<String, Double> map = new TreeMap<String, Double>();
```
D `Map<String, Double> map = new HashMap<>();`

E `Map<String, Double> map = new TreeMap<>();`

F `Map<String, Double> map = new Map<>();`

G All of the above.

H None of the above.

Question 2

What is the value of the following expression?

```
1 + "2"+ 3 + 4 * 5 + 6
```

A `123456`
B `21`
C `42`
D `"1229"`
E `"123206"`
F `"123456"`
G `"1241"`
H `"1266"`
I Some other value.
J The value of this expression is undefined.
K This expression would result in a compilation error.

Question 2

What is the value of the following expression?

```
1 + "2"+ 3 + 4 * 5 + 6
```

A `123456`
B `21`
C `42`
D `"1229"`
E `"123206"`
F `"123456"`
G `"1241"`
H `"1266"`
I Some other value.
J The value of this expression is undefined.
K This expression would result in a compilation error.

Question 2: Reasoning

```
1 + "2"+ 3 + 4 * 5 + 6
```

- The `*` operator has higher precedence than than `+`

Question 2: Reasoning

```
1 + "2"+ 3 + 4 * 5 + 6
```

- The `*` operator has higher precedence than than `+`
- `1 + "2"+ 3 + 20 + 6`

Question 2: Reasoning

```
1 + "2"+ 3 + 4 * 5 + 6
```

- The `*` operator has higher precedence than than `+`
- `1 + "2"+ 3 + 20 + 6`
- All the `+` operators have equal precedence, so now we evaluate from left to right.

Question 2: Reasoning

```
1 + "2"+ 3 + 4 * 5 + 6
```

- The `*` operator has higher precedence than than `+`
- `1 + "2"+ 3 + 20 + 6`
- All the `+` operators have equal precedence, so now we evaluate from left to right.
- `"12" + 3 + 20 + 6`

Question 2: Reasoning

```
1 + "2"+ 3 + 4 * 5 + 6
```

- The `*` operator has higher precedence than than `+`
- `1 + "2"+ 3 + 20 + 6`
- All the `+` operators have equal precedence, so now we evaluate from left to right.
- `"12" + 3 + 20 + 6`
- `"123" + 20 + 6`

Question 2: Reasoning

```
1 + "2"+ 3 + 4 * 5 + 6
```

- The `*` operator has higher precedence than than `+`
- `1 + "2"+ 3 + 20 + 6`
- All the `+` operators have equal precedence, so now we evaluate from left to right.
- `"12" + 3 + 20 + 6`
- `"123" + 20 + 6`
- `"12320"+ 6`

Question 2: Reasoning

```
1 + "2"+ 3 + 4 * 5 + 6
```

- The `*` operator has higher precedence than than `+`
- `1 + "2"+ 3 + 20 + 6`
- All the `+` operators have equal precedence, so now we evaluate from left to right.
- `"12" + 3 + 20 + 6`
- `"123" + 20 + 6`
- `"12320"+ 6`
- `"123206"`

Question 3

Which methods can access the protected members of a class?

A Only static methods of the same class.
B Only final methods of the same class.
C Only protected methods of the same class.
D Only private and protected methods of the same class.
E Only methods defined in the same class.
F Only methods defined in the same package (including those in the class itself).
G Only methods within the same class or its children.
H Only methods within the same class, its children, or in the same package.

Question 3

Which methods can access the protected members of a class?

A Only static methods of the same class.
B Only final methods of the same class.
C Only protected methods of the same class.
D Only private and protected methods of the same class.
E Only methods defined in the same class.
F Only methods defined in the same package (including those in the class itself).
G Only methods within the same class or its children.
H Only methods within the same class, its children, or in the same package.

Question 4

Fill in the blanks to make a method that takes two arguments, a `Collection` of `String` objects and a `String` to search for, and returns a collection containing only the strings found in the input collection that contain the search string somewhere in them. The method should be able to be called without an instance of the containing class. You may assume that the arguments are not null and that the Collection does not contain any null objects. Do not modify the collection passed in as an argument.

Question 11

```
public BLANK BLANK
  filterStrings(BLANK strs,
                String search) {

  Collection<String> result = new BLANK;

  for(BLANK) {
    BLANK

    if(s.contains(search)) {
      BLANK
    }
  }

  return BLANK;
}
```

Question 11

```
public static BLANK
  filterStrings(BLANK strs,
                String search) {

  Collection<String> result = new BLANK;

  for(BLANK) {
    BLANK

    if(s.contains(search)) {
      BLANK
    }
  }

  return BLANK;
}
```

Question 11

```
public static Collection<String>
  filterStrings(BLANK strs,
                String search) {

  Collection<String> result = new BLANK;

  for(BLANK) {
    BLANK

    if(s.contains(search)) {
      BLANK
    }
  }

  return BLANK;
}
```

Question 11

```
public static Collection<String>
  filterStrings(Collection<String> strs,
                String search) {

  Collection<String> result = new BLANK;

  for(BLANK) {
    BLANK

    if(s.contains(search)) {
      BLANK
    }
  }

  return BLANK;
}
```

Question 11

```
public static Collection<String>
  filterStrings(Collection<String> strs,
                String search) {

  Collection<String> result = new ArrayList<>();

  for(BLANK) {
    BLANK

    if(s.contains(search)) {
      BLANK
    }
  }

  return BLANK;
}
```

Question 11

```
public static Collection<String>
  filterStrings(Collection<String> strs,
                String search) {

  Collection<String> result = new ArrayList<>();

  for(BLANK) {
    BLANK

    if(s.contains(search)) {
      BLANK
    }
  }

  return result;
}
```

Question 11

```
public static Collection<String>
  filterStrings(Collection<String> strs,
                String search) {

  Collection<String> result = new ArrayList<>();

  for(String s : strs) {
    BLANK

    if(s.contains(search)) {
      BLANK
    }
  }

  return result;
}
```

Question 11

```
public static Collection<String>
  filterStrings(Collection<String> strs,
                String search) {

  Collection<String> result = new ArrayList<>();

  for(String s : strs) {
    BLANK

    if(s.contains(search)) {
      result.add(s);
    }
  }

  return result;
}
```

Question 11

```
public static Collection<String>
  filterStrings(Collection<String> strs,
                String search) {

  Collection<String> result = new ArrayList<>();

  for(String s : strs) {
    /* no code here */

    if(s.contains(search)) {
      result.add(s);
    }
  }

  return result;
}
```