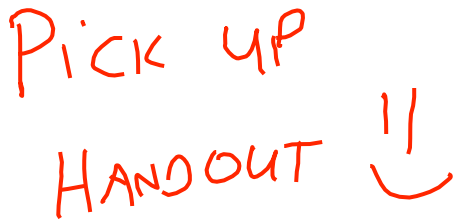
Variables and UML



What can be improved? (variables)

```
public void goDirection(String directionName) {
  boolean wentToRoom = false;
  for (Direction direction : currentRoom.getDirections()) {
    if (direction.getDirectionName().equalsIgnoreCase(directionName)) {
      wentToRoom = true;
      currentRoom = direction.getDestinationRoom();
      break; return;
    }
  }
  if (!wentToRoom) {
    System.out.println("I can't go " + directionName);
  }
}
```

A) Eliminating temporary variable
B) Eliminating intermediate results
C) Eliminating control flow variable
D) Shrinking scope of variable
E) Prefer write once variable

What can be improved? (variables)

```
public static void main(String[] args) {
    String currRoomName = "";
    Boolean continuePlaying = true;

    // deals with args ...

    Layout mapLayout = UserInterface.LoadMap(newMapUrl);
    Map<String, Room> playMap = GameState.GenerateVirtualMap(mapLayout);
    Room currRoom = playMap.get(mapLayout.getStartingRoom());

    while (continuePlaying) {
        currRoomName = GameState.play(currRoom);
        currRoom = playMap.get(currRoomName);

        if (currRoomName.equals("EXIT")) {
            continuePlaying = false;
        }
    }
}
```

true

String

break;

A) Eliminating temporary variable
B) Eliminating intermediate results
C) Eliminating control flow variable
D) Shrinking scope of variable
E) Prefer write once variable

What can be improved?

A) Eliminating temporary variable
B) Eliminating intermediate results
C) Eliminating control flow variable
D) Shrinking scope of variable
E) Prefer write once variable

```
String description;
String currentRoom = layout.getStartingRoom();
String done = "";
for (int i = 0; i < layout.getRooms().length; ) {
    int currentRoomIndex = layout.getRoomFromName(currentRoom);
    description = layout.getRooms()[currentRoomIndex].getDescription();
    System.out.println(description);
    ArrayList<String> directionName = new ArrayList<String>();
    for (int j = 0;
         j < layout.getRooms()[currentRoomIndex].getDirections().length;
         j++) {
       directionName.add(layout.getRooms()[currentRoomIndex]
          .getDirections()[j].getDirectionName().toLowerCase());
    }

    String direction = getDirectionsOption(directionName);
```

What can be improved? (variables)

```
public static void checkFloorPlan() throws Exception {
    // ...  (removed stuff)

    for (Room currRoom : roomCollection.values()) {
        boolean roomFound = false;

        for (Direction currDirection : currRoom.getDirections()) {
            roomFound = roomFound || findRoomInConnecting(currRoom.getName(),
                    roomCollection.get(currDirection.getRoom()));
        }

        if (!roomFound) {
            throw new BadFloorPlanJsonException("Rooms not connected.");
        }
    }
}
```

A) Eliminating temporary variable
B) Eliminating intermediate results
C) Eliminating control flow variable
D) Shrinking scope of variable
E) Prefer write once variable

Which is better?

A

```
String originalDirectionName = input.substring(3);
return "I can't go " + originalDirectionName + "\n";
```

B

```
return "I can't go " + input.substring(3) + "\n";
```

Which is better?

A

```
String input = scanner.nextLine();
String output = gameController.handleInput(input);
```

B

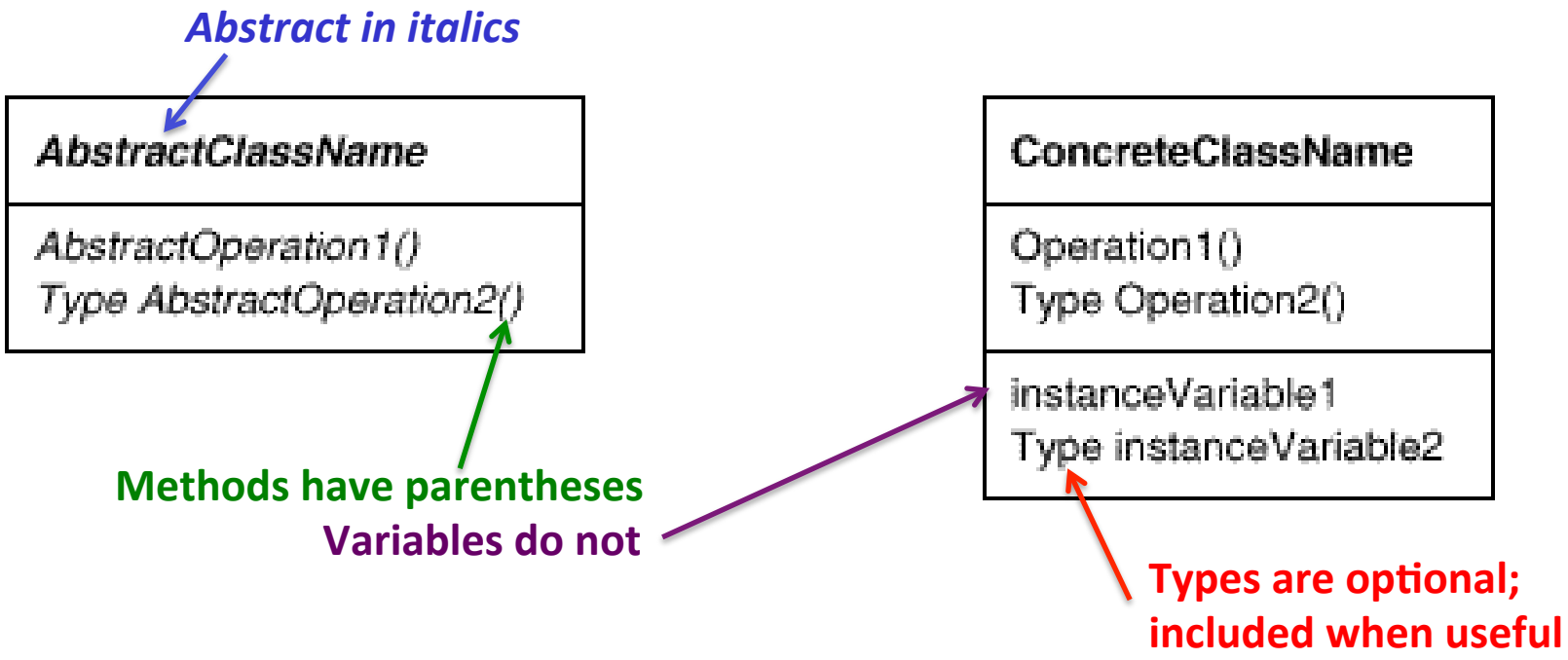
```
String output = gameController.handleInput(scanner.nextLine());
```

UML Class Diagrams

- **Unified Modeling Language (UML)**
  - A standard for diagrammatic representations in software engineering.
- **The Class Diagram is the main building block for object-oriented modeling; it shows:**
  - the system's classes
  - their attributes and operations (or methods), and
  - the relationships among objects

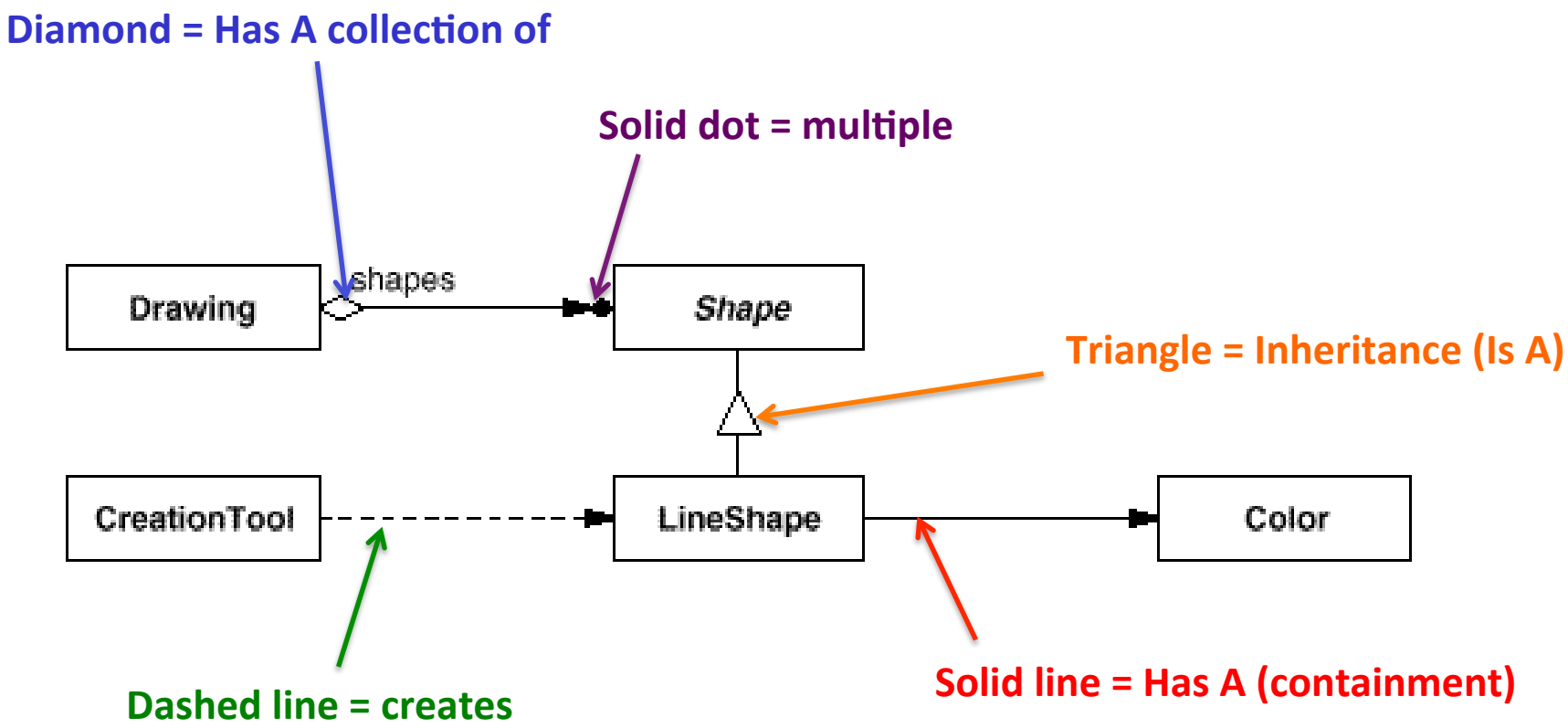
Class/Object Notation

- **Class definitions**



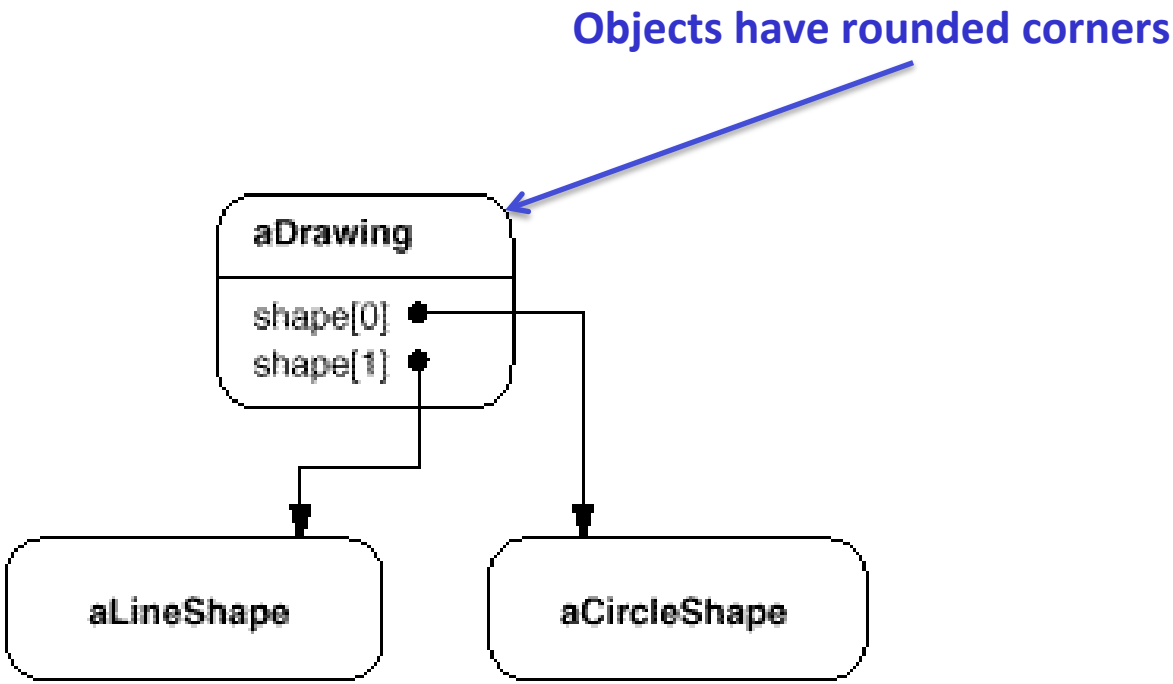
Class/Object Notation (cont.)

- Class relationships

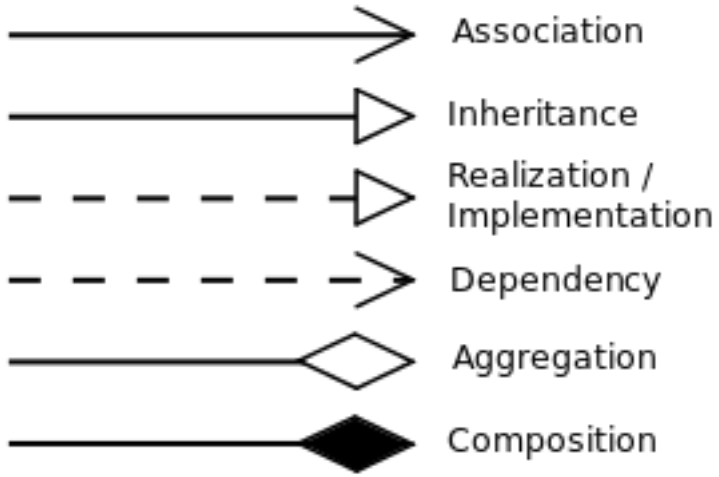


Class/Object Notation (cont.)

- **Object instances**



Relationships



Scrabble


GAME
WORD
SCRABBLE
DOUBLE LETTER SCORE
DOUBLE WORD SCORE
TRIPLE LETTER SCORE

Scrabble word score

- **Sum of the letter values**

English-language editions of Scrabble contain 100 letter tiles, in the following distribution:

- 2 blank tiles (scoring 0 points)
- 1 point: E ×12, A ×9, I ×9, O ×8, N ×6, R ×6, T ×6, L ×4, S ×4, U ×4.
- 2 points: D ×4, G ×3.
- 3 points: B ×2, C ×2, M ×2, P ×2.
- 4 points: F ×2, H ×2, V ×2, W ×2, Y ×2.
- 5 points: K ×1.

Scrabble word score, continued

```
public static int wordScore(String word) {
   int score = 0;
   for (int i = 0 ; i < word.length() ; i++) {
     char letter = word.charAt(i);
     score += letterScore(letter);
   }
   return score;
}
```

how would you implement this?

1) map
2) switch statement
3) integer array

Control-flow based

```
public static int letterScore(char c) {
     char upperC = Character.toUpperCase(c);
     switch (upperC) {
     case 'A':
     case 'E':
     case 'I':
     case 'L':
     case 'N':
     case 'O':
     case 'R':
     case 'S':
     case 'T':
     case 'U':
               return 1;
     case 'D':
     case 'G':
               return 2;
     case 'B':
     case 'C':
     case 'M':
     case 'P':
               return 3;
     case 'F':
     case 'H':
     case 'V':
     case 'W':
     case 'Y':
               return 4;
     case 'K':
               return 5;
     case 'J':
     case 'X':
               return 8;
     case 'Q':
     case 'Z':
               return 10;
     default:
              // handle error
     }
     // should never reach here
     return 0;
}
```

Table-based Solution

```
private static final int [] scoresByChar =
  {/* A */ 1, /* B */ 3,  /* C */ 3, /* D */ 2, /* E */ 1,
   /* F */ 4, /* G */ 2,  /* H */ 4, /* I */ 1, /* J */ 8,
   /* K */ 5, /* L */ 1,  /* M */ 3, /* N */ 1, /* O */ 1,
   /* P */ 3, /* Q */ 10, /* R */ 1, /* S */ 1, /* T */ 1,
   /* U */ 1, /* V */ 4,  /* W */ 4, /* X */ 8, /* Y */ 4,
   /* Z */ 10};

public static int letterScore2(char c) {
  char cAsUppercase = Character.toUpperCase(c);
  int index = cAsUppercase - 'A';
  if (index < 0 || index >= 26) {
    // handle error
  }
  return scoresByChar[index];
}
```