Возможные пути выполнения фрагмента кода

```
String evaluate_salary_and_return_name(Employee e)
{
  if (e.title() == "CEO" || e.salary() > 100000)
  {
    std::cout << e.name() << " " << e.surname() << " is overpaid.\n";
  }
  else
  {
    std::cout << e.name() << " is not overpaid.\n";
  }

  return e.name() + " " + e.surname();
}
```

а) Пути без ошибок:
1. Под if true
2. Под if false, то есть путь else.

б) Где могли сгенерироваться исключения:
1. В конструкторе Employee, когда создавалась копия.
2. В функции-члене title.
3. В == при преобразованиях или непосредственно работе оператора.
4-5. Аналогично 2-3, но уже для salary и >.
6. В || при при преобразованиях или непосредственно работе оператора.
7. В функции-члене name
8. В операторе вывода, если результат e.name() является пользовательским классом.
9-10. Аналогично 7-8, но для surname.
11-12. Если оказалось, что пошли по пути else, то в точности 7-8.
13. e.name() уже работала без ошибок, но оператор вывода мог испортить данные, поэтому снова есть шанс, что произойдет ошибка.
14. В + при преобразованиях или непосредственно работе оператора.
15-16. Аналогично 13-14, только для surname, которая при этом могла еще ни разу не вызываться (по пути else), то есть изменение объекта не является обязательным для возможности ошибки.
17. В return, если получившимся после преобразований типом будет не String, а преобразование в String идет с ошибкой.
18. В деструкторе копии класса-аргумента, если программист все же рискнул добавить туда исключения или просто аварийно завершил программу, обнаружив, что инвариант нарушен.

Итого: 20 путей.