Null Dereferenciations

1. Null Dereferenciation, intra procedural

```
1  public class Example1
2  {
3          public static void main(String[] args)
4          {
5                  String str=null;
6                  int x;
7                  if(args.length>0)
8                          str=args[args.length-1];
9                  x=str.length();
10         }
11 }
```

2. Null Dereferenciation, inter procedural

```
1  public class Example2_1
2  {
3          public static void main(String[] args)
4          {
5                  int a=args.length+1;
6                  int b=args.length;
7                  int x;
8                  String str=Example2_2.foo(a,b);
9                  x=str.length();
10         }
11 }
```

```
1  public class Example2_2
2  {
3          public static String foo(int a, int b)
4          {
5                  String str=null;
6                  if(b!=0)
7                          str=Float.toString((float) a/b);
8                  return str;
9          }
10 }
```

3. Models

```
1  package lib;
2
3  import com.facebook.infer.builtins.InferUndefined;
4  import com.facebook.infer.builtins.InferBuiltins;
5
6
7  public class ClassFromLib {
8
9          public int doSomething()
10         {
11                 int x = InferUndefined.int_undefined();
12                 InferBuiltins.assume(x > 0);
13                 return x;
14         }
15
16 }
```

```java
import lib.ClassFromLib;

public class Example3
{
        public static void main(String[] args)
        {
                int x,y;
                String str=null;
                ClassFromLib c=new ClassFromLib();
                x=c.doSomething();
                if(x>0)
                        str="abc";
                y=str.length();
        }
}
```

4. Allowing null as input

```java
import javax.annotation.Nullable;

public class Example4
{
        public int getLength(@Nullable String str)
        {
                return str.length();
        }
}
```

5. Null Dereferenciation: inter procedural large scale:
   The following Android code snippets show a bug which was found in the the source code of duckduckgo
   Source:
   https://code.facebook.com/posts/1537144479682247/finding-inter-procedural-bugs-at-scale-with-infer-static-analyzer/
   last access 21.11.2017

```java
865 public void feedItemSelected(String feedId) {
866         FeedObject feedObject = DDGApplication.getDB().selectFeedById(feedId);
867         feedItemSelected(feedObject);

483 public FeedObject selectFeedById(String id){
484         FeedObject out = null;
485         Cursor c = null;
486         try
487         {
488                 c = this.db.query(FEED_TABLE, null, "_id=?", new String[]{id},
489                 if (c.moveToFirst()) {
490                         out = getFeedObject(c);
491                 }
492         } finally {
493         if(c!=null) {
494             c.close();
495       }
496   }
497   return out;
498 }
```

```
842 public void feedItemSelected(FeedObject feedObject) {
843  // keep a reference, so that we can reuse details while saving
844     DDGControlVar.currentFeedObject = feedObject;
845     DDGControlVar.mDuckDuckGoContainer.sessionType = SESSIONTYPE.SESSION_FEED;
846      String url = feedObject.getUrl();
```

Check for nullness of method's in and output

6. assigning null to non nullable field

```
1  import javax.annotation.Nullable;
2
3  public class Example6
4  {
5          private String str;
6
7          public Example6()
8          {
9                  this.str="";
10         }
11
12         public void foo(@Nullable String s)
13         {
14                 this.str=s;
15         }
16 }
```

7. field which is not annotated with nullable is not initialized

```
1  import javax.annotation.Nullable;
2
3  public class Example7
4  {
5          private String str;
6
7          public Example7()
8          {
9          }
10 }
```

8. return null although not annotated as nullable

```
1  import javax.annotation.Nullable;
2
3  public class Example8
4  {
5          public String foo(int a)
6          {
7                  String s=a>0?"Test":null;
8                  return s;
9          }
10 }
```

Resource Leaks

9. input stream without closure

```
1  import java.io.FileInputStream;
2  import java.io.IOException;
3
4  public class Example9
5  {
6          public void foo(String str) throws IOException
7          {
8                  FileInputStream in=new FileInputStream(str);
9          }
10 }
```

10. input stream with not always reachable closure

```
1  import java.io.FileInputStream;
2  import java.io.IOException;
3
4  public class Example10
5  {
6          public int foo(String str) throws IOException
7          {
8                  int ret;
9                  FileInputStream in=new FileInputStream(str);
10                 ret=in.read();
11                 in.close();
12                 return ret;
13         }
14 }
```

Memory leaks in C++

11. Memory leak should be detected

```
1  void foo()
2  {
3          int* p=new int;
4          //do something
5  }
```

12. No memory leak should be detected

```
1  int* bar()
2  {
3          int* p=new int;
4          return p;
5  }
```