ECS 140A Homework 2

Jones, Hardy
999397426
Professor Olsson
Winter 2014

1. Give the output using (a) static scoping and (b) dynamic scoping.

   (a) With static scoping, the program outputs
   ```
   Q 1 1
   P 1 3
   Q 1 5
   P 1 3
   ```
   (b) With dynamic scoping, the program outputs
   ```
   Q 1 1
   P 1 1
   Q 5 5
   P 5 5
   ```

2. Consider the following program.

   (a) Show the runtime stack, including static pointers, each time a block or procedure is entered or exited.

action	stack	static pointers
start:	(empty)	
+A:	A	
+P/D:	A P	P→A
+S:	A P S	P→A S→P
+Q/C:	A P S Q	P→A S→P Q→P
+T:	A P S Q T	P→A S→P Q→P T→Q
+R/B:	A P S Q T R	P→A S→P Q→P T→Q R→A
+U:	A P S Q T R U	P→A S→P Q→P T→Q R→A U→R
-U:	A P S Q T R	P→A S→P Q→P T→Q R→A
-R/B:	A P S Q T	P→A S→P Q→P T→Q
-T:	A P S Q	P→A S→P Q→P
-Q/C:	A P S	P→A S→P
-S:	A P	P→A
-P/D:	A	
+E:	A E	E→A
+R/F:	A E R	E→A R→A
+U:	A E R U	E→A R→A U→R
-U:	A E R	E→A R→A
-R/F:	A E	E→A
-E:	A	
-A:	(empty)	

(b) Show the runtime stack and the display each time a block or procedure is entered or exited.

action	stack	display (@ = pointer to or address of)
start:	(empty)	(empty)
+A:	A	@A
+P/D:	A P	@A @P
+S:	A P S	@A @P @S
+Q/C:	A P S Q	@A @P @Q
+T:	A P S Q T	@A @P @Q @T
+R/B:	A P S Q T R	@A @R
+U:	A P S Q T R U	@A @R @U
-U:	A P S Q T R	@A @R
-R/B:	A P S Q T	@A @P @Q @T
-T:	A P S Q	@A @P @Q
-Q/C:	A P S	@A @P @S
-S:	A P	@A @P
-P/D:	A	@A
+E:	A E	@A @E
+R/F:	A E R	@A @R
+U:	A E R U	@A @R @U
-U:	A E R	@A @R
-R/F:	A E	@A @E
-E:	A	@A
-A:	(empty)	(empty)

3. Give the output from the following program for each of the following parameter passing combinations:

(a) x, y, z by value.

```
1 2 3
1 2 3
2 1 1
1 2 3
```

(b) x, y, z by reference.

```
1 2 3
2 1 1
1 2 2
1 1 1
```

(c) x, y, z by name.

```
1 2 3
2 2 1
1 2 2
1 1 1
```

(d) x by reference, y by name, z by value.

```
1 2 3
2 2 1
1 2 2
1 2 1
```

(e) x, y by name, z by reference.

```
1 2 3
2 2 1
1 2 2
1 1 1
```

4. Consider the following Ada-like program.

   Suppose OOPS is raised at A for each invocation of Q.

   (a) Assume the code in handler H1 and H2 do not execute a raise statement. For the invocation of Q at I1, which handler (H1, H2, or H3) will be executed? Explain.

   H1 will be executed as it is the closest exception handler to the point of exception.

   (b) Assume the code in handler H1 and H2 do not execute a raise statement. For the invocation of Q through the invocation of R at I2, which handler (H1, H2, or H3) will be executed? Explain.

   H1 will be executed as it is the closest exception handler to the point of exception.

   (c) Assume now that H1 executes a raise OOPS statement (but H2 doesn't). Explain what happens for each of the invocations of I1 and I2.

   For I1, when Q raises OOPS, H1 attempts to handle the exception. H1 then raises OOPS, so exception control goes up the call stack to H3. Assuming H3 doesn't raise OOPS as well, it handles the exception.

   For I2, when Q raises OOPS, H1 attempts to handle the exception. H1 then raises OOPS, so exception control goes up the call stack to H2. H2 doesn't raise OOPS, so it handles the exception.

   (d) Assume now that H1 and H2 each executes a raise OOPS statement. Explain what happens for each of the invocations of I1 and I2.

   For I1, when Q raises OOPS, H1 attempts to handle the exception. H1 then raises OOPS, so exception control goes up the call stack to H3. Assuming H3 doesn't raise OOPS as well, it handles the exception.

   For I2, when Q raises OOPS, H1 attempts to handle the exception. H1 then raises OOPS, so exception control goes up the call stack to H2. H2 attempts to handle the exception. H2 then raises OOPS, so exception control goes up the call stack to H3. Assuming H3 doesn't raise OOPS as well, it handles the exception.