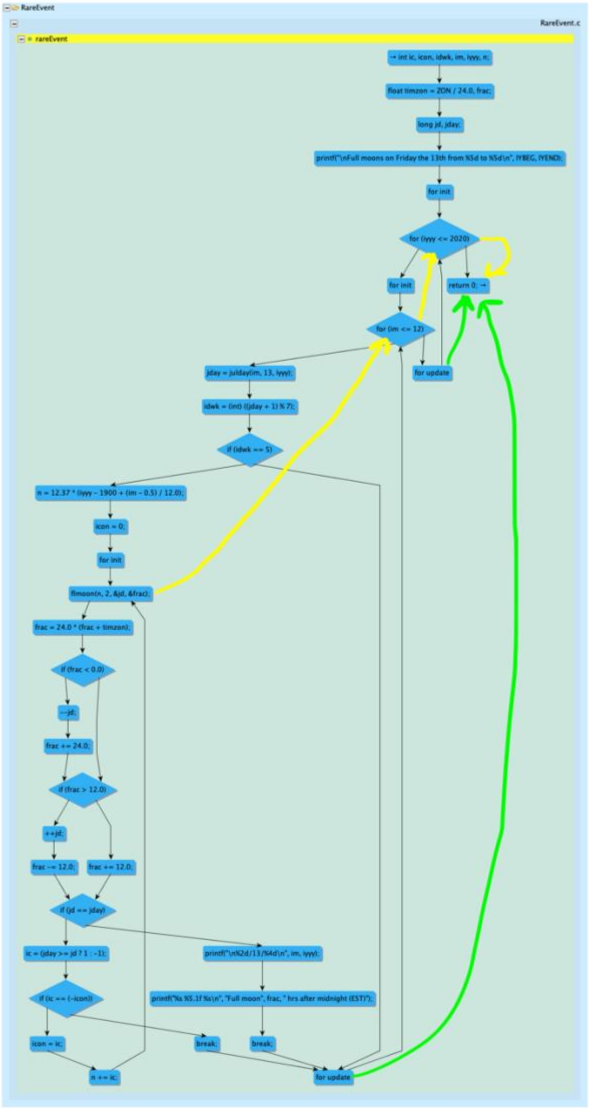
Stamatios Morellas
SE 421 - Assignment 5 Write-up
10/19/20

Q3. Identify the correct successor for each loop in function rareEvent from RareEvent.c on the given cfg (rareEventCFG.png) and show the corresponding continuation edges. Include the CFG and your work in the writeup. (3 points)



RareEvent
RareEvent.c
rareEvent
→ int ic, icon, idwk, im, iyyy, n;
float timzon = ZON / 24.0, frac;
long jd, jday;
printf("\nFull moons on Friday the 13th from %5d to %5d\n", IYBEG, IYEND);
for init
for (iyyy <= 2020)
for init
return 0; →
for (im <= 12)
for update
jday = julday(im, 13, iyyy);
idwk = (int) ((jday + 1) % 7);
if (idwk == 5)
n = 12.37 * (iyyy - 1900 + (im - 0.5) / 12.0);
icon = 0;
for init
flmoon(n, 2, &jd, &frac);
frac = 24.0 * (frac + timzon);
if (frac < 0.0)
--jd;
frac += 24.0;
if (frac > 12.0)
++jd;
frac -= 12.0;
frac += 12.0;
if (jd == jday)
ic = (jday >= jd ? 1 : -1);
printf("\n%2d/13/%4d\n", im, iyyy);
if (ic == (-icon))
printf("%s %5.1f %s\n", "Full moon", frac, " hrs after midnight (EST)");
icon = ic;
break;
break;
n += ic;
for update

Figure 1: Markup CFG for rareEvent function. Yellow lines represent link to successor node and green lines represent continuation edges.

Q4. Write a sequence of queries to mark all the back edges yellow and make them dashed. Include in your writeup the sequence of queries and the resulting CFG. (2 points)

Queries Used:

```
var rareEvent = functions("rareEvent")
var rareEventCFG = cfg(rareEvent)

var rareEventControlFlowBackEdges = rareEventCFG.edges(XCSG.ControlFlowBackEdge).retainEdges()

var m = new Markup

m.setEdge(rareEventControlFlowBackEdges, MarkupProperty.EDGE_STYLE, MarkupProperty.LineStyle.DASHED)

m.setEdge(rareEventControlFlowBackEdges, MarkupProperty.EDGE_COLOR, Color.YELLOW)

show(rareEventCFG, m)
```

Result:

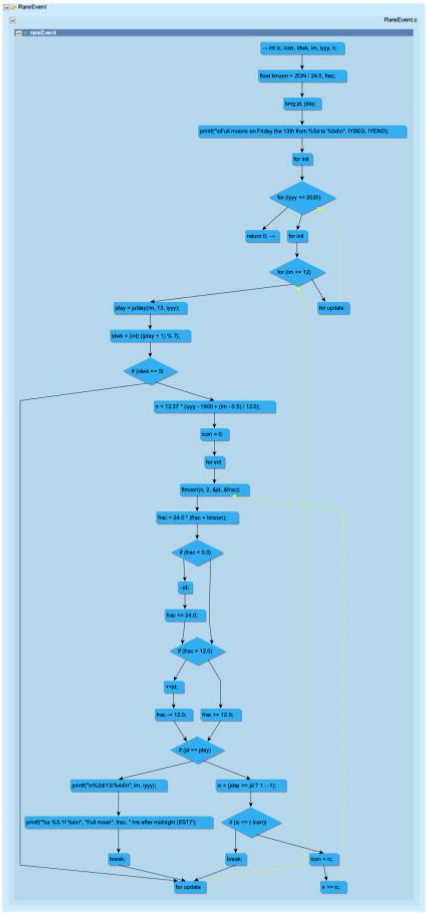

RareEvent
RareEvent.c
rareEvent
→ int ic, icon, idwk, im, iyyy, n;
float timzon = ZON / 24.0, frac;
long jd, jday;
printf("\nFull moons on Friday the 13th from %5d to %5d\n", IYBEG, IYEND);
for init
for (iyyy <= 2020)
return 0; →
for init
for (im <= 12)
jday = julday(im, 13, iyyy);
for update
idwk = (int) ((jday + 1) % 7);
if (idwk == 5)
n = 12.37 * (iyyy - 1900 + (im - 0.5) / 12.0);
icon = 0;
for init
flmoon(n, 2, &jd, &frac);
frac = 24.0 * (frac + timzon);
if (frac < 0.0)
--jd;
frac += 24.0;
if (frac > 12.0)
++jd;
frac -= 12.0;
frac += 12.0;
if (jd == jday)
printf("\n%2d/13/%4d\n", im, iyyy);
ic = (jday >= jd ? 1 : -1);
printf("%s %5.1f %s\n", "Full moon", frac, " hrs after midnight (EST)");
if (ic == (-icon))
break;
break;
icon = ic;
for update
n += ic;

Figure 2

Q5. Write a sequence of queries to exclude all the back edges. Include in your writeup the sequence of queries and the CFG without the back edges. (2 points)

Queries Used:

```
var rareEvent = functions("rareEvent")
var rareEventCFG = cfg(rareEvent)

var rareEventControlFlowBackEdges = rareEventCFG.edges(XCSG.ControlFlowBackEdge)

var removeEdges = rareEventCFG.differenceEdges(rareEventControlFlowBackEdges)

show(removeEdges)
```

Result:

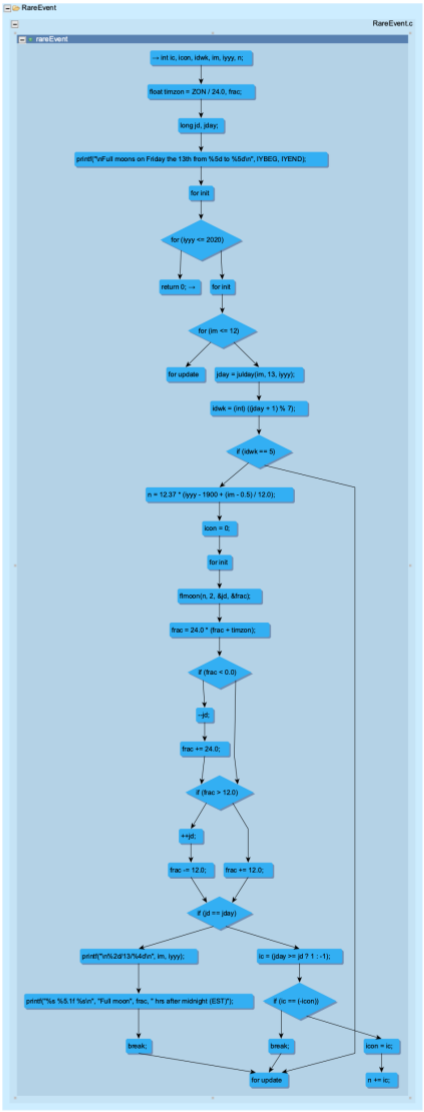
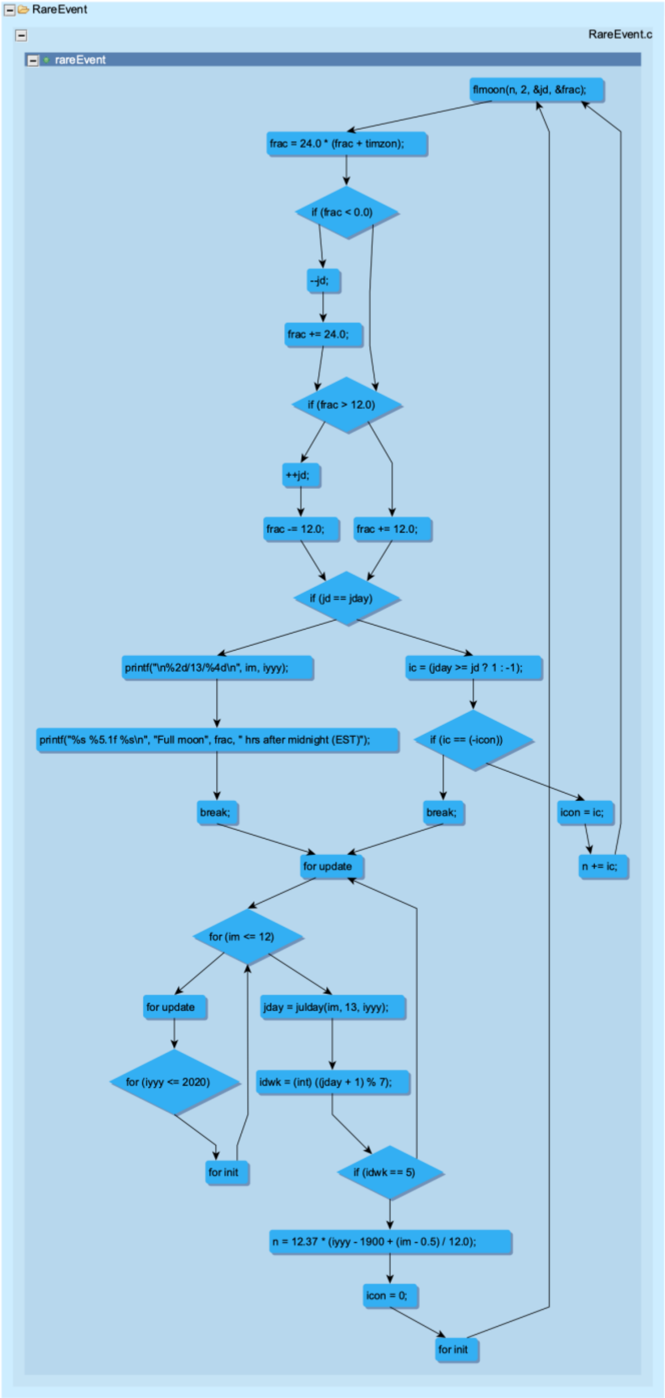

RareEvent
RareEvent.c
rareEvent
→ int ic, icon, idwk, im, iyyy, n;
float timzon = ZON / 24.0, frac;
long jd, jday;
printf("\nFull moons on Friday the 13th from %5d to %5d\n", IYBEG, IYEND);
for init
for (iyyy <= 2020)
return 0; →
for init
for (im <= 12)
for update
jday = julday(im, 13, iyyy);
idwk = (int) ((jday + 1) % 7);
if (idwk == 5)
n = 12.37 * (iyyy - 1900 + (im - 0.5) / 12.0);
icon = 0;
for init
flmoon(n, 2, &jd, &frac);
frac = 24.0 * (frac + timzon);
if (frac < 0.0)
--jd;
frac += 24.0;
if (frac > 12.0)
++jd;
frac -= 12.0;
frac += 12.0;
if (jd == jday)
printf("\n%2d/13/%4d\n", im, iyyy);
ic = (jday >= jd ? 1 : -1);
printf("%s %5.1f %s\n", "Full moon", frac, " hrs after midnight (EST)");
if (ic == (-icon))
break;
break;
icon = ic;
for update
n += ic;

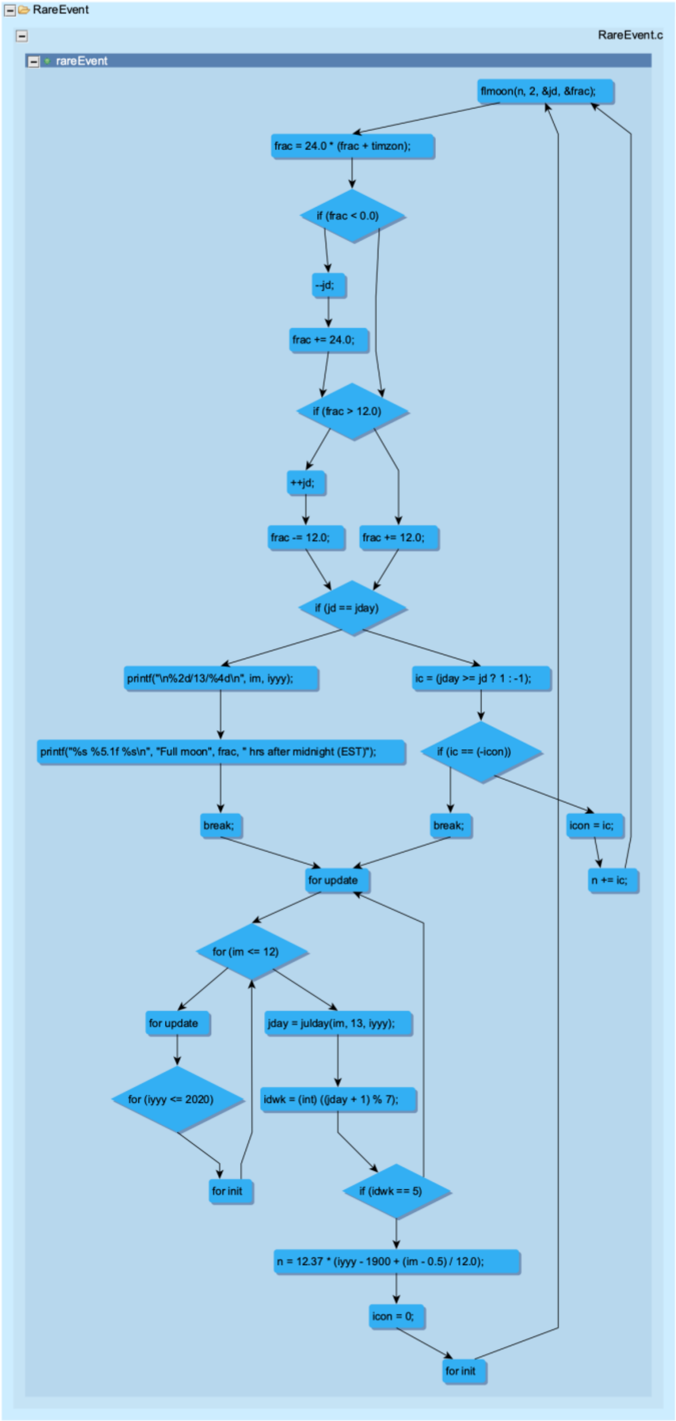
Figure 3

Q6. Use the sequence of Atlas queries in (firstTry-loopbody-queries.txt) on the three loops A, B, and C in function rareEvent in RareEvent.c. Show in your writeup the resultant loop body for the one loop where the sequence of queries works correctly. (2 points)

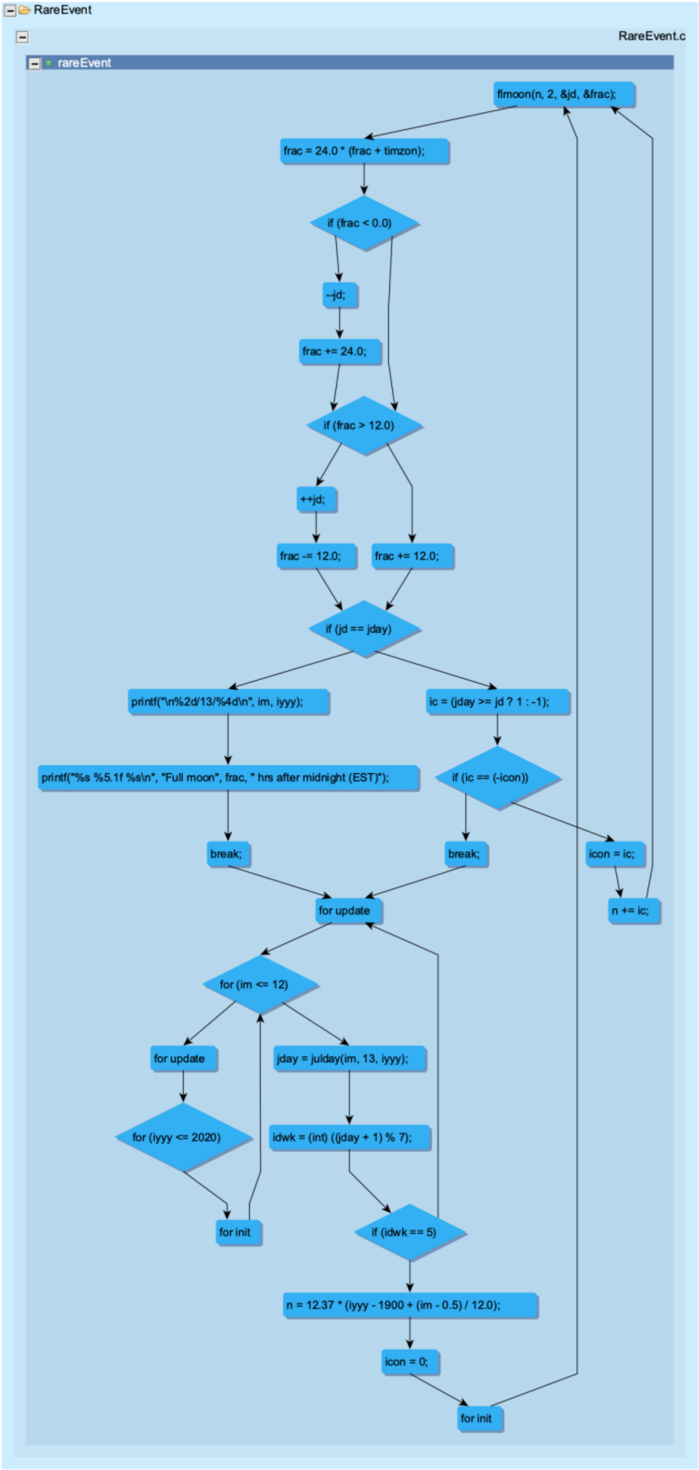
For Loop A:


RareEvent
RareEvent.c
rareEvent
flmoon(n, 2, &jd, &frac);
frac = 24.0 * (frac + timzon);
if (frac < 0.0)
--jd;
frac += 24.0;
if (frac > 12.0)
++jd;
frac -= 12.0;
frac += 12.0;
if (jd == jday)
printf("\n%2d/13/%4d\n", im, iyyy);
ic = (jday >= jd ? 1 : -1);
printf("%s %5.1f %s\n", "Full moon", frac, " hrs after midnight (EST)");
if (ic == (-icon))
break;
break;
icon = ic;
n += ic;
for update
for (im <= 12)
for update
jday = julday(im, 13, iyyy);
for (iyyy <= 2020)
idwk = (int) ((jday + 1) % 7);
for init
if (idwk == 5)
n = 12.37 * (iyyy - 1900 + (im - 0.5) / 12.0);
icon = 0;
for init

For Loop B:


RareEvent
RareEvent.c
rareEvent
flmoon(n, 2, &jd, &frac);
frac = 24.0 * (frac + timzon);
if (frac < 0.0)
--jd;
frac += 24.0;
if (frac > 12.0)
++jd;
frac -= 12.0;
frac += 12.0;
if (jd == jday)
printf("\n%2d/13/%4d\n", im, iyyy);
ic = (jday >= jd ? 1 : -1);
printf("%s %5.1f %s\n", "Full moon", frac, " hrs after midnight (EST)");
if (ic == (-icon))
break;
break;
icon = ic;
n += ic;
for update
for (im <= 12)
for update
jday = julday(im, 13, iyyy);
for (iyyy <= 2020)
idwk = (int) ((jday + 1) % 7);
for init
if (idwk == 5)
n = 12.37 * (iyyy - 1900 + (im - 0.5) / 12.0);
icon = 0;
for init

For Loop C:


RareEvent
RareEvent.c
rareEvent
flmoon(n, 2, &jd, &frac);
frac = 24.0 * (frac + timzon);
if (frac < 0.0)
--jd;
frac += 24.0;
if (frac > 12.0)
++jd;
frac -= 12.0;
frac += 12.0;
if (jd == jday)
printf("\n%2d/13/%4d\n", im, iyyy);
ic = (jday >= jd ? 1 : -1);
printf("%s %5.1f %s\n", "Full moon", frac, " hrs after midnight (EST)");
if (ic == (-icon))
break;
break;
icon = ic;
n += ic;
for update
for (im <= 12)
for update
jday = julday(im, 13, iyyy);
for (iyyy <= 2020)
idwk = (int) ((jday + 1) % 7);
for init
if (idwk == 5)
n = 12.37 * (iyyy - 1900 + (im - 0.5) / 12.0);
icon = 0;
for init

After running the queries given by firstTry-loopbody-queries.txt, I noticed that all of the output CFGs generated from selecting each individual loop-back edge from each respective loop in the rareEvent function are the same. You can see each of the outputs shown above. Although I was confused at first as to what to make of this, I later concluded that the only loop which the sequence of queries works correctly for is **loop A**.

Q7. The Atlas queries in (firstTry-loopbody-queries.txt) would not work correctly on two of the loops. In your writeup, identify those loops, show for each loop the resultant loop body, and describe in one sentence your understanding of the cause for incorrect computation of the loop body. (3 points)

The two loops that the Atlas queries found in (firstTry-loopbody-queries.txt) would not work correctly for are **loops B** and **C**. My understanding as to why this doesn't produce the correct result is that the provided queries only give the loop body of the outermost loop and not the other two loops which are nested inside loop A.

Q8. Write succinct pseudo code for the algorithm to compute the loop body of the inner loops identified in #5. To do so modify the sequence of Atlas queries in (firstTry-loopbody-queries.txt) that works correctly for non-nested loops but not for an inner loop. Use the notion of *loop depth* and *Outer loops* as discussed in the lecture and exemplified in the companion slides. Give the pseudo in your write up (4 points)

```
1. Create a variable for the "rareEvent" function.
2. Create a cfg of the "rareEvent" function.
3. Show the cfg
4. Select back edge
5. Select the node representing the head of the loop body.
6. Get the node(s) of the tail end(s) of the loop body.
7. Compute the loop body
8. Show the output of the loop body.
```

Q9. Revise the given sequence of Atlas queries in (firstTry-loopbody-queries.txt) so that the revised sequence would work correctly on the inner loops identified in #5. Include in your writeup the correct result for the inner loops identified in #5. (4 points)

```
var rareEvent = functions("rareEvent")
var rareEventCFG = cfg(rareEvent)
show(rareEventCFG)
var backEdge = selected.eval().edges().one()
var loopHeader = backEdge.to()
var tail = edges(XCSG.LoopChild).forward(toQ(loopHeader)).retainNodes
var loopBody = tail.induce(edges(XCSG.ControlFlow_Edge))
show(loopBody)
```

Q10. Modify the given sequence of Atlas queries in (firstTry-loopbody-queries.txt) to add the continuation edge and delete the back edge for each of the inner loops identified in #5. Include the sequence of queries and show the result in your writeup. (5 points)

Couldn't figure this one out......