

COMP11 - HW1 Sample Answers

Ex 1-2e

This statement is true, because one could simply take the desired 4-character program and add an R command to the end, which would result in no graphical change. One could go on to say that where n is an integer less than m , any shape made from n characters could also be made from m characters, because any number of R commands could be appended.

====

This statement is true because the five letter sequences can simply be thought of as each of the four letter sequences with an R or an F added to the end. This explains why the number of sequences doubles every time the length of the sequence increases by 1. If we are just adding an R to the end of each of the 16 sequences, the shape will not change so each of the different shapes will be present in the 5 character shapes. This idea can be generalized to say "Every shape you can draw with an X letter long program can also be drawn with an $(X+1)$ letter long program"

====

Since the command "R" doesn't actually draw anything, a four-letter program can be made into a five-letter program by simply adding an R to the end. An x -letter program can draw a y -letter program simply by adding $x - y$ "R"s to the end of the code, as long as $x > y$, because adding "R"s to the END of a program doesn't affect the resulting shape.

====

Any four letter program can be written as a five letter program by adding an R on to the end, because this only changes the final direction of the tractor and does not draw any additional lines. In general, any shape can be created using more letters by simply adding more Rs to the end of the program. This is not the only way of doing it, but it is the simplest.

====

Every four letter program can also be written as a five letter program. The simplest explanation is that the five letter program can copy the four letter program and then end with R. Ending with R does not draw a different picture, so any design done in a 4 letter program can be done in a 5 letter one as well. This statement can be generalized: any image made by m letters can be made by $m+1$ letters because the program with $(m+1)$ letters can simply copy the smaller program, and the $(m+1)$ th letter can simply be R to not make a different image.

Ex 1-3

a. a square

It draws a square because of two things. One, the program M makes a line and turns. This creates one side of the square. The program M has "M" at the end which makes it repeat. Since the program repeats endlessly, it is endlessly creating sides and turns, thus making a square (or, rather, squares... millions of them)

b. It makes no pattern

The program makes nothing because as the program M starts, it is instructed to execute program M, which is instructed to execute program M, which is instructed to execute program M, which is basically it goes straight down a rabbit hole/abyss and never gets to the FR, which would have drawn something.

====

a. a four-sided box

When the code executes, the previously defined abbreviation of M is followed: the tractor goes forward (F), turns right (R), and then gets to M, thus causing the tractor to go forward and turn right again before returning to M. This program effectively instructs the tractor to drive forward and turn right alternately until the processor times out (explaining the CPU time exceeded error). This creates a four-sided box, each side of length 1, that has been traced over numerous times.

b. blank

As with the program in part A, the program looks to the definition for the M abbreviation when it executes. In this case, however, the first statement in the definition is M itself, causing the program to return to the beginning of the definition repeatedly. The code never executes past this first function. As a result, no picture is drawn--the tractor is never told to move or turn.

=====

- a. a square with sides of length one that would have infinite retraces if the program did not stop.

The assignment for M is FRM. Since M is part of the assignment for the char M, the program will return to the assignment to substitute its value in. Therefore, the program will continue to substitute FRM forever. FR comes before M in the assignment, so the program will create a line with a right turn at the end (90 deg clockwise rotation) before substituting in the value for M. Thus, the program will be continuously drawing a square as it will always be plugging in another M and then once again, defining it.

- b. a blank picture.

Unlike the previous assignment, the assignment for M is MFR, which puts M before FR. Thus, the program reads M before FR and performs no visible action. Since it reads M first, it will substitute in the assignment for M, which is MFR. This will only lead it to read another M, so it will again substitute a value in for M. Since the program will never substitute in a sequence that has a char other than M first, the program will be stuck in an infinite loop of substituting in assignments for M. Because it never reads a command other than M, and does nothing other than substitute in values for M, there will be nothing but a blank picture following the program's execution.

=====

- a. A square

The pattern that the first mystery code describes is a square. This is an infinite loop, with infinite number of retraces. The program starts with an assignment of M to FRM, when M is called the first thing tractor will do to draw a vertical line upwards. Then, the R will change the direction of the next line to horizontal to the right. The third program is M itself, which by assignment is equal to FR. Then the tractor executes the F which describes a horizontal line. This time the R will also change the direction of the next line to vertical downward. And Tractor reaches M again and draws a vertical line downwards. Finally, it executes R. and draw horizontal line from right to left.

- a. Nothing

The second mystery code does not draw anything, this is because it is also an infinite loop but one that does not include any drawing instructions. The F and R are outside the infinite loop. In principle, tractor should finish the loop and then executes the F and the R. but since the loop is infinite, then nothing is drawn. The loop is infinite because the first thing that is executed when M is called is M. Every time M is called it starts with calling itself before drawing anything.

=====

- a. a square.

Every time M appears, it is replaced by FRM because M is defined as FRM in the abbreviation. Thus, the program executes a series of code that looks like (FR(FR(FR(FR(FR(FR.....))))). The program stops after 10000 actions by default. The "show seq" button indicates that there are 5000 forward and 5000 right commands. A series of FR commands makes a square that keeps retracing itself because each line segment is perpendicular to the last one (in the clockwise direction).

- b. blank.

Every time M appears, it is to be replaced by (MFR) by definition. Thus, the program keeps substituting MFR in for M. The "meat" of the code, the FR sequence, is never executed because the program keeps substituting in for M until the maximum number of tasks the program can do is reached. The FR part of the code is sadly never executed because MFR keeps replacing M. The poor little code never stood a chance.

=====

a. A rotational and clockwise route of a square started from the left side

Every time M is running, FR is run and following with M, then M is running again and FR is run again since M is the abbreviation of FRM. The whole procedure would repeat infinitely

b. a blank picture

Every time M is running, M is run again following with FR because M is the abbreviation of MFR, then M is run second time following with FR again, FR is left behind all the time since the second command will not execute until the first is completed. The whole procedure would then repeat infinitely.

Ex 1-4 b : Why 25 is the Shortest Program

In the picture, there are 16 lines, so those must each be drawn at least once. There are 8 outer corners. One additional turn is required to switch from drawing the vertical lines to drawing the horizontal lines. ($16+8+1=25$)

====

The 25 character program is the shortest possible program because it utilizes only right handed turns. There is a definite number of Fs that need to be used because of the number of lines that compose the figure. The use of Rs is the variable that can truly alter the length of the code (unless one was retracing and in that case the use of F's could also make the program longer). By using only right handed turns, the number of R's is minimized and as such the program is as short as it can be.

====

The shape must necessarily contain 16 tractor forward movements as well as 9 R turns if one wants to avoid retraces. No retraces and no extra turns would logically produce the shortest sequence.

====

This is the shortest code length because in order to create the shape there needs to be at least 1 F for every line in the drawing, of which there are 16. There also needs to be at least one R for every corner of which there are 8. Another R is needed to switch between the two rectangular parts. The minimum length program would use exactly 1 R for every corner. Thus, the minimum program length is $16+8+1=25$.

Ex 1-4c+d: Short Programs with Abbreviations and Loops

Here, sorted by program length, are many solutions. All work. Points were awarded based on how short the program is. The shortest program submitted was 9 characters long.

```

9      4FRFRFRF
10     4 3FR: RR 4RF
11     AB4FR:RF.4B:
11     ABFR. 4RF4B:
11     ABRF. 4BFBBF
11     AN4FR:RF. 4N
12     ABFR2FFR:B.B
12     AMFR 2 FRF: M. M
13     AB4FR:. 3BRF: B
13     AB4FR:RRFRB.B
13     ABFFRFRFR. 4B
13     ABFRFRFRF. 4B
13     ABRFRFRFR. 4B
13     ALFRFRFRF. 4L
14     AC2FRFRFF. CRFC
14     AC2FRFRFF. CRFC
14     AL2FRFRFF. LRFL
14     AS 2FFRFRF. SFRS
14     AS2FFRFRF. SFRS
15     2FRFRFRFRFRFRF
15     AB2FRFRFF:. BRFB
15     ABFRFRF. BBRFB
15     ABFRFRFF. BBRFB
15     ABFRFRFF. BBRFB
15     ABFRFRFF. BBRFB
15     ABFRFRFF. BBRFB
15     ACFFRFRF. CCFRCC

```

```

15  ADFRFRFF.DDRFDD
15  AE FRFRFF. EE R F EE
15  AIFRFRFF.IIRFII
15  AMFRFRFF.MMRFMM
15  AS FRFRFF. SS RF SS
15  AS22FR:FF:.SRFS
15  ASFRFRFF.SSRFSS
16  AB 2FR: F. 2BF: R 2FB
16  ADFRFRF. 2DF: R 2FD
16  AX2FR:F. X FX FR 2FX
17  AB2FR.2BFF:RF2BFF
17  AB2FR:F. B F B FRF B F B
17  ABFRFRF. BFBFRFBFB
17  ABFRFRF. BFBFRFBFB
17  ABFRFRF. BFBFRFBFB
17  ABFRFRF.BFBFRFBFB
17  ACFRFRF.CFCFRFCFC
17  ADFRFRF. DDFRFDFFD
17  AL3R:. AB4FR:FL. 4B
17  ALFRFRF. LFLFRFLFL
17  AMFRFRF.MFMFRFMFM
17  AQ2FR. 2QFF:RF2QFF
17  ASF2RF:. S F S F RF S F S
17  ASFRFRF. SFSFRFSFS
17  AW3FR. ABRRW. WFW3B
17  AWRFRFRF. WFRFRFWFW
17  AXFRFRF. XFXFRFXFX
18  ACFRFRF.RCFRCFRFCFC
18  ASFRFRF. RSFSFRFSFS
18  AWRFRFRRRR. 4W: R 4FR
19  AB FR FR FF. BB RF BF RF RF
19  AB FRF. AC B RFF B RF. C B C
19  ABFRFRFFFRFRF.BFRFB
19  AC FRFRF. AS C F C. S FR F S
19  AC2FR. CFFCFFRFFCFFCF
19  ACFRFRF.ABCFC.BFRFB
19  AS2FR. SFFSFFRFSFFSF
19  ASFR. ALSSF. LF LS FL FL
19  ATFRFRF. FTFTRRRTFTF
19  AX2FR:FFF2RF:.XFRFX
19  AXFR. AZXXF. ZFZXFZFZ
19  AZ2FR:3F:2RF:. ZFRFZ
20  AB RF. AC BBFF BB. BCF BF C
20  AB2FR:.BFFBFFRFBFFBF
20  ABFRFRF. ACBF. 2C: R F C B
20  ABRF. ACFFBB. CC2FB:BC
20  ADFRFR. D FF D FF R F D FF D F
20  AZFRFRF.2FRZ:ZFZFRFZ
21  ABFR. BBFFBBFBFFBFFBF
21  ABRFFFFR.ATFRF.TBTTTBT
21  AXFRF.AYR3F:R.XY3X:YX
22  ABFF. ACRF. AD2BCC:. DFRD
22  ABRF.ACFB.3B:FCBCCBFCB
22  ABRF.ACFF2B.RR3B:CFR2C
22  ASRF. RSSSFFSSFSFSFFSS
23  3R:2F2RF:F:RF2FR:3F:2RF
23  AB2FR:F.AC3R:.BBRF2BC:B
23  ABFR. ACFFR. BB F C B CC BF C BF
23  ABFRFR. BFRRRBFFBFRFBFF
23  ALRRRF. F 2FR: FF 3FR: 2FFLL
23  ASFR. ADFS. FD S FD S D D S FD S F
23  ATFR.FFTTFTFTTFFTTTRRTF
24  2FR:3F:2RF:FR2F:2RF:F3FR
24  AB2FR:F.AT3R:.3FR:F3BT:B
24  ALFFFRF. LRLRFRRRFRFRLRFF
25  ABFFFRF.RBRBRFRFRFRBRFF
25  AX FF. AE R FRF X RF RF. X E 3R: F E F
25  FRFRFFFFRFRFRFRFRFRFRFRFRF
26  ACRF. ADRFF. 4C:DC 2CD: FCDD CC
26  ATFFF. APRFR. TPTPFRFRFPFF
26  AXFRFRFRRR. AYFRFRF. XXX YY RF
26  AYFF.AZFR.YZZZZZYRRYZZZF

```

27 AXFR. AYFX. AZFY. 2X:ZX2Y:XZXF
28 ABFR.ACFF. FBBBFBBCCBBFBFBBCBF
28 ABFRFRF. ACRRR. AXBCBCBC. XBBRF
29 AB FR. AC FB. AD FC. AE CBDB. 3B: ECEF
30 F 2RF: RRRF 3RF: RRRF F 2RRRF: F 3RF: F
32 ALRRR. AYFR. LYY 3F: RYF 3FL: 3F: 2LF: F
32 AXFR.AYFF.AZXXYXXF. APXF. AQZXFZ. Q