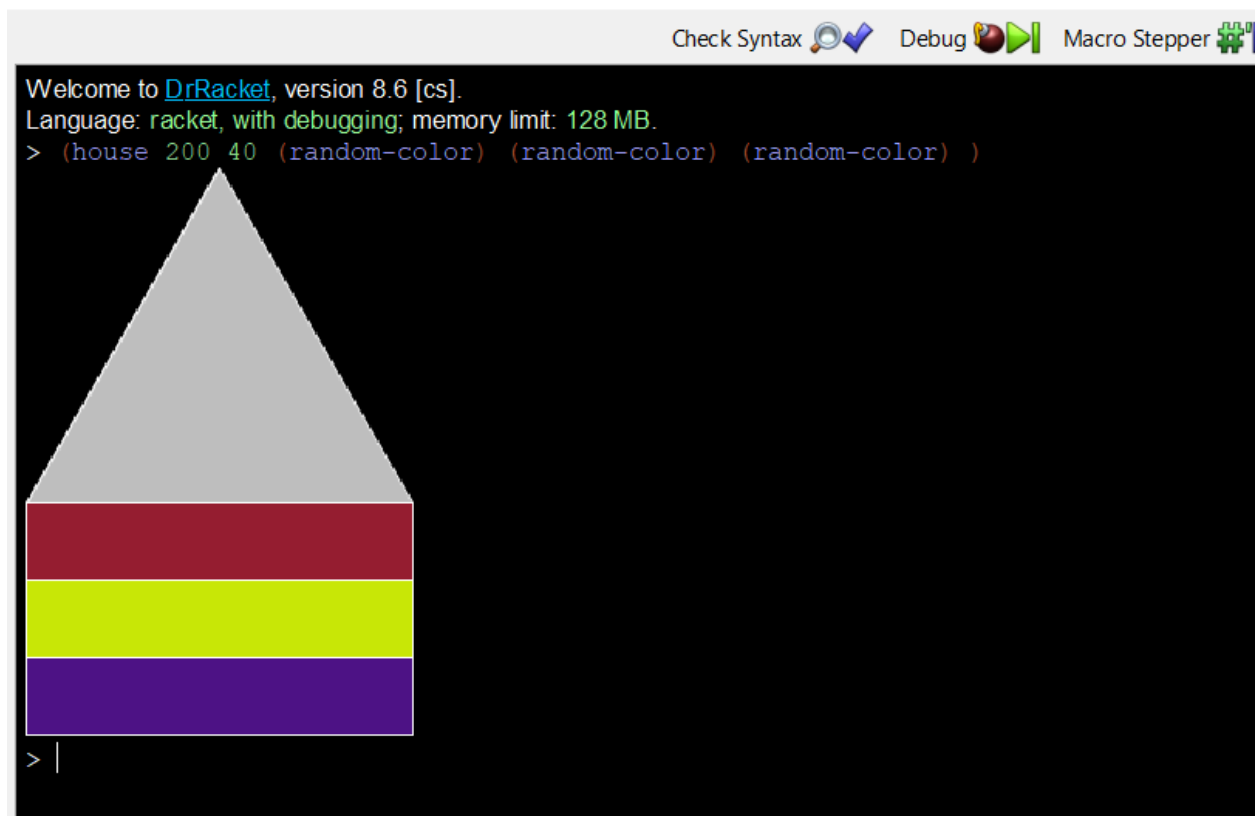


Racket Programming Assignment 2: Racket Functions and Recursion

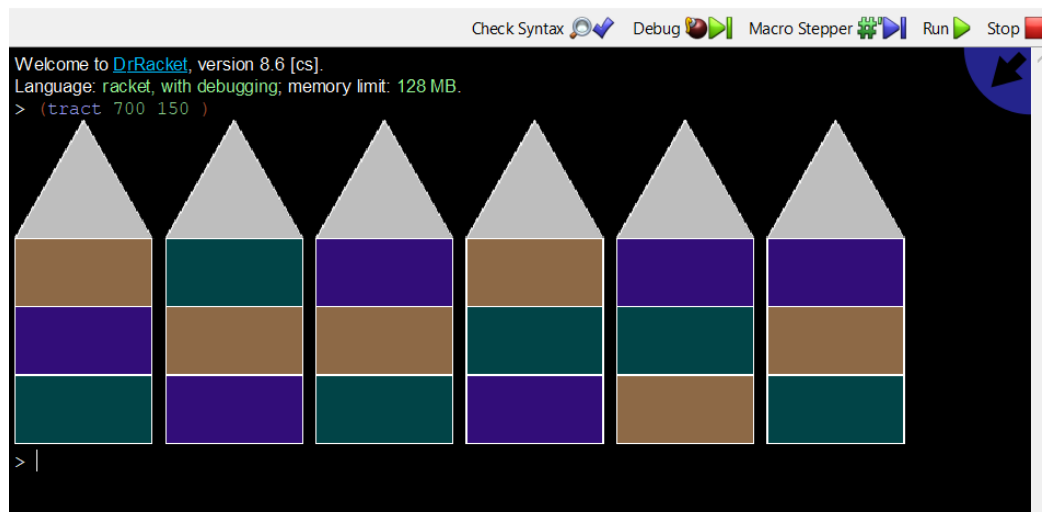
Learning Abstract: This assignment contains programs which generate images using the 2htdp/images library, mostly using but not restricted to recursion.

Task 1 Colorful Permutations of Tract Houses:

Demo for House:



Demo for Tract:



Code:

Task 1.rkt - DrRacket

File Edit View Language Racket Insert Scripts Tabs Help

Task 1.rkt (define ...)

```
1 #lang racket
2 (require 2htdp/image)
3 (define (random-color) (color (rgb-value) (rgb-value) (rgb-value) ))
4 (define (rgb-value) (random 256) )
5 (define (house width height floor1 floor2 floor3)
6   (above
7     (overlay
8       (triangle width "outline" "white" )
9       (triangle width "solid" "grey" )
10    )
11    (overlay
12      (rectangle width height "outline" "white")
13      (rectangle width height "solid" floor1)
14    )
15    (overlay
16      (rectangle width height "outline" "white")
17      (rectangle width height "solid" floor2)
18    )
19    (overlay
20      (rectangle width height "outline" "white")
21      (rectangle width height "solid" floor3)
22    )
23  )
24 )
25
26 (define (tract width height)
27   (define width-of-house( / ( - width 100 ) 6 ) )
28   (define height-of-floor( / height 3 ) )
29   (define color1 (random-color) )
30   (define color2 (random-color) )
31   (define color3 (random-color) )
32   (define space (square 10 "solid" "black") )
33   (beside
34     (house width-of-house height-of-floor color3 color1 color2 )
35     space
36     (house width-of-house height-of-floor color2 color3 color1)
37     space
38     (house width-of-house height-of-floor color1 color3 color2)
39     space
40     (house width-of-house height-of-floor color3 color2 color1)
41     space
42     (house width-of-house height-of-floor color1 color2 color3)
43     space
44     (house width-of-house height-of-floor color1 color3 color2)
45   )
46 )
```

Determine language from source





Windows taskbar with search bar and application icons.

Task 2 Dice

Demos:

—

□

Check Syntax  Debug  Macro Stepper  Run 

```
Welcome to DrRacket, version 8.6 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (roll-dice)
4
> (roll-dice)
2
> (roll-dice)
1
> (roll-dice)
1
> (roll-dice)
2
> (roll-for-1)
3 4 6 6 5 3 2 1
> (roll-for-1)
6 6 2 1
> (roll-for-1)
4 2 3 6 6 1
> (roll-for-1)
6 6 6 6 2 6 5 2 5 3 5 1
> (roll-for-1)
3 1
> (roll-for-11)
6 2 5 3 2 4 3 1 3 5 4 3 1 1
> (roll-for-11)
1 3 5 2 3 2 1 3 6 4 3 1 4 6 3 5 2 4 2 4 2 2 1 1
> (roll-for-11)
1 3 2 6 6 3 5 2 3 6 6 5 2 3 4 3 5 4 5 4 2 5 1 5 2 3 6 2 4 5 3 4 2 4 4 1 1
> (roll-for-11)
3 5 3 6 6 5 4 4 5 2 3 2 4 4 3 6 4 6 2 3 2 6 3 1 5 2 1 1
> (roll-for-11)
3 5 2 3 3 3 3 6 6 1 5 4 3 1 3 6 3 1 1
> |
```

Check Syntax  Debug  Macro Stepper  Run

```
Welcome to DrRacket, version 8.6 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (roll-for-odd-even-odd)
4 4 2 3 5 3 3 2 1
> (roll-for-odd-even-odd)
2 3 4 2 3
> (roll-for-odd-even-odd)
4 6 3 4 2 4 4 3
> (roll-for-odd-even-odd)
4 6 5 3 3 5 3 2 1
> (roll-for-odd-even-odd)
3 5 4 5
> (roll-two-dice-for-a-lucky-pair)
(6 6)
> (roll-two-dice-for-a-lucky-pair)
(5 3) (2 6) (2 3) (6 4) (6 1)
> (roll-two-dice-for-a-lucky-pair)
(1 6)
> (roll-two-dice-for-a-lucky-pair)
(4 6) (1 6)
> (roll-two-dice-for-a-lucky-pair)
(1 2) (1 1)
> (roll-two-dice-for-a-lucky-pair)
(1 5) (4 2) (3 3)
> (roll-two-dice-for-a-lucky-pair)
(2 6) (2 6) (5 2)
> (roll-two-dice-for-a-lucky-pair)
(2 2)
> (roll-two-dice-for-a-lucky-pair)
(4 1) (5 2)
> (roll-two-dice-for-a-lucky-pair)
(3 3)
> (roll-two-dice-for-a-lucky-pair)
(5 1) (6 4) (6 2) (1 3) (3 3)
> |
```

```
1 #lang racket
2 (define (roll-dice) ( + (random 6) 1 ) )
3
4 (define (roll-for-1)
5   (define die (roll-dice) )
6   (display die ) (display " ")
7   (cond
8     ( ( not (eq? die 1) ) )
9       (roll-for-1)
10    )
11  )
12 )
13 (define (roll-for-11)
14   (roll-for-1)
15   (define die (roll-dice) )
16   (display die) (display " ")
17   ( cond
18     ( ( not (eq? die 1) ) )
19       (roll-for-11)
20     )
21   )
22 )
23
24 (define (roll-for-even)
25   (define die (roll-dice) )
26   (display die ) (display " ")
27   (cond
28     ( ( not (even? die ) ) )
29       (roll-for-even)
30     )
31   )
32 )
33
34 (define (roll-for-odd)
35   (define die (roll-dice) )
36   (display die ) (display " ")
37   (cond
38     ( ( not (odd? die ) ) )
39       (roll-for-odd)
40     )
41   )
42 )
43
```

Code: Determine language from source▼

```

43
44 (define (roll-for-odd-even-odd) (roll-for-odd) (roll-for-even) (roll-for-odd) )
45
46 (define (roll-two-dice-for-a-lucky-pair)
47   (define die1 (roll-dice))
48   (define die2 (roll-dice))
49   (define total-7 ( =(+ die1 die2 ) 7 ) )
50   (define total-11 ( = (+ die1 die2 ) 11) )
51   (define pairs ( = die1 die2 ) )
52   (display "(")
53   (display die1)
54   (display " ")
55   (display die2)
56   (display ")")
57   (cond
58     ( (not (or total-7 total-11 pairs) )
59       (roll-two-dice-for-a-lucky-pair) ) ) )
60
61
62
63
64

```

Task 3 Number Sequences

Demo:

Check Syntax
Debug
Macr

```

Welcome to DrRacket, version 8.6 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (square 5 )
25
> (square 10 )
100
> (sequence square 15)
1 4 9 16 25 36 49 64 81 100 121 144 169 196 225
> (cube 2 )
8
> (sequence cube 15 )
1 8 27 64 125 216 343 512 729 1000 1331 1728 2197 2744 3375
> (triangular 2)
3
> (triangular 1)
1
> (sequence triangular 20)
1 3 6 10 15 21 28 36 45 55 66 78 91 105 120 136 153 171 190 210
> |

```

Code:

```
1 | #lang racket
2
3 | (define (square n )
4 |   ( * n n ) )
5
6 | (define (cube n)
7 |   ( * n n n ) )
8
9 | (define (sequence name n )
10 |   (cond
11 |     ( ( = n 1 )
12 |       (display ( name 1 ) ) (display " " )
13 |     )
14 |     (else
15 |       (sequence name ( - n 1 ) )
16 |       (display ( name n ) ) (display " ")
17 |     )
18 |   )
19 | )
20
21 | (define (triangular n )
22 |   (cond
23 |     ( ( = n 1 ) 1 )
24 |     ( ( > n 1 )
25 |       ( + n ( triangular ( - n 1 ) ) )
26 |     )
27 |   )
28 | )
29
30
```

Task 4 Hirst Dots


```
Welcome to DrRacket, version 8.6 [cs].  
Language: racket, with debugging; memory limit: 128 MB.  
> (first-dots 4)
```



```
>
```

Demo:

—

□

×

Check Syntax  Debug  Macro Stepper  Run  Stop 

Welcome to [DrRacket](#), version 8.6 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (first-dots 10)



>

Code:

Task4.rkt - DrRacket

File Edit View Language Racket Insert Scripts Tabs Help

Task4.rkt (define ...)

```
1: Task 1.rkt × | 2: Task2.rkt × | 3: Task3.rkt × | 4: Task4.rkt × +
1 | #lang racket
2 | (require 2htdp/image)
3 | (define (random-color) ( color (rgb-value) (rgb-value) (rgb-value) ) )
4 | (define (rgb-value) (random 256) )
5 | (define (space) (square 20 "solid" "black") )
6 | (define (dot) (circle 30 "solid" (random-color) ) )
7 |
8 | (define (row-of-dots n)
9 |   (cond
10 |     ( ( = n 0 )
11 |       empty-image
12 |     )
13 |     ( ( > n 0 )
14 |       (beside (row-of-dots ( - n 1 ) ) (dot) (space) )
15 |     )
16 |   )
17 | )
18 |
19 | (define (rect-of-dots r c)
20 |   (cond
21 |     ( ( = r 0 )
22 |       empty-image
23 |     )
24 |     ( ( > r 0 )
25 |       (above (rect-of-dots ( - r 1 ) c) (row-of-dots c) )
26 |     )
27 |   )
28 | )
29 |
30 | (define (hirst-dots n )
31 |   (rect-of-dots n n) )
32 |
```

Task 5 Chanelling Frank Stella

Demo:



The screenshot shows the DrRacket IDE interface. At the top right, there is a 'Check Syntax' button with a magnifying glass and a checkmark icon. Below this, a small window with a '+' icon contains the text: 'Welcome to [DrRacket](#), version 8.6 [cs]. Language: racket, with debugging; memory limit: 128 MB.' The main workspace has a black background and displays two Stella objects. The first object, created with the command `(Stella 50 5 "red")`, is a small red circle with five concentric black outlines. The second object, created with the command `(Stella 100 10 (random-color))`, is a larger purple circle with ten concentric black outlines. A prompt character `>` is visible at the bottom left of the workspace.

```
Welcome to DrRacket, version 8.6 [cs].  
Language: racket, with debugging; memory limit: 128 MB.  
> (Stella 50 5 "red" )  
  
> (Stella 100 10 (random-color) )  
  
>
```

Code:

Task5.rkt - DrRacket

File Edit View Language Racket Insert Scripts Tabs Help

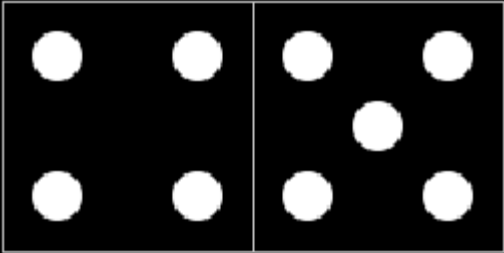
Task5.rkt (define ...)

1: Task 1.rkt × | 2: Task2.rkt × | 3: Task3.rkt × | 4: Task4.rkt × | 5: Task5.rkt

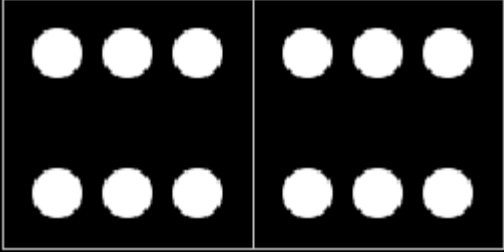
```
1 #lang racket
2 (require 2htdp/image)
3
4 (define (random-color) (color (rbg-value) (rbg-value) (rbg-value) ) )
5 (define (rbg-value) (random 256 ) )
6
7 (define (Stella radius count color)
8   (define unit ( / radius count ) )
9   (paint-nested-circles-one 1 count unit color ) )
10
11 (define (paint-nested-circles-one from to unit color )
12   (define radius ( * from unit ) )
13   (cond
14     ( ( = from to )
15       (framed-circle radius color)
16     )
17     ( ( < from to )
18       (overlay
19         (framed-circle radius color )
20         (paint-nested-circles-one ( + from 1 ) to unit color )
21       )
22     )
23   )
24 )
25
26 (define (framed-circle radius color)
27   (overlay
28     (circle radius "outline" "black")
29     (circle radius "solid" color)
30   )
31 )
```

Task 6 Dominos

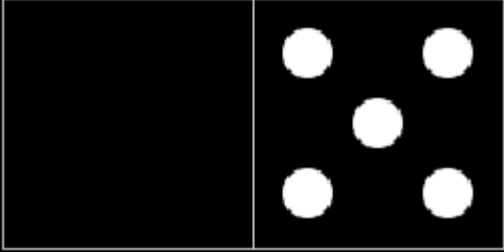
```
Welcome to DrRacket, version 8.6 [
Language: racket, with debugging; m
> (domino 4 5 )
```



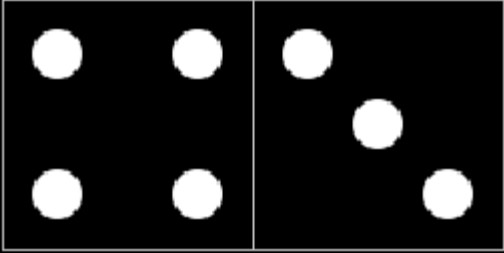
```
> (domino 6 6)
```



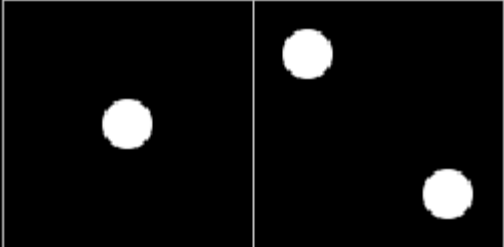
```
> (domino 0 5 )
```


--

```
> (domino 4 3 )
```



```
> (domino 1 2 )
```



Demo:

Code:

```
1: Task 1.rkt  × | 2: Task2.rkt  × | 3: Task3.rkt  × | 4: Task4.rkt  × | 5: Task5.rkt  >
1 #lang racket
2 (require 2htdp/image)
3
4 ;problem parameters
5 ;-variables to denote the side of a tile and the dimensions of a pip
6
7 (define side-of-tile 100)
8 (define diameter-of-pip ( * side-of-tile 0.2 ) )
9 (define radius-of-pip ( / diameter-of-pip 2 ) )
10
11 ;Numbers used for offsetting pips from the center of a tile
12 ;- d and nd are used as offsets in the overlay/offset function applications
13
14 (define d ( * diameter-of-pip 1.4 ) )
15 (define nd ( * -1 d ) )
16
17 ;The blank tile and pip generator
18 ;- Bind one variable to a blank tile and another to a pip
19
20 (define blank-tile (square side-of-tile "solid" "black" ) )
21 (define (pip) (circle radius-of-pip "solid" "white" ) )
22
23 ;The basic tiles
24 ; Bind one variable to each of the basic tiles
25
26 (define basic-tile1 (overlay (pip) blank-tile) )
27
28 (define basic-tile2
29   (overlay/offset (pip) d d
30     (overlay/offset (pip) nd nd blank-tile)
31   )
32 )
33
34 (define basic-tile3 (overlay (pip) basic-tile2 ) )
35
36 (define basic-tile4
37   (overlay/offset (pip) d d
38     (overlay/offset (pip) d nd
39       (overlay/offset (pip) nd d
40         (overlay/offset (pip) nd nd blank-tile)
41       )
42     )
43   )
44 )
45 )
46 )
47 )
48 )
49 )
50 )
51 )
52 )
53 )
54 )
55 )
56 )
57 )
58 )
59 )
60 )
61 )
62 )
63 )
64 )
65 )
66 )
67 )
68 )
69 )
70 )
71 )
72 )
73 )
74 )
75 )
76 )
77 )
78 )
79 )
80 )
81 )
82 )
83 )
84 )
85 )
86 )
87 )
88 )
89 )
90 )
91 )
92 )
93 )
94 )
95 )
96 )
97 )
98 )
99 )
100 )
101 )
102 )
103 )
104 )
105 )
106 )
107 )
108 )
109 )
110 )
111 )
112 )
113 )
114 )
115 )
116 )
117 )
118 )
119 )
120 )
121 )
122 )
123 )
124 )
125 )
126 )
127 )
128 )
129 )
130 )
131 )
132 )
133 )
134 )
135 )
136 )
137 )
138 )
139 )
140 )
141 )
142 )
143 )
144 )
145 )
146 )
147 )
148 )
149 )
150 )
151 )
152 )
153 )
154 )
155 )
156 )
157 )
158 )
159 )
160 )
161 )
162 )
163 )
164 )
165 )
166 )
167 )
168 )
169 )
170 )
171 )
172 )
173 )
174 )
175 )
176 )
177 )
178 )
179 )
180 )
181 )
182 )
183 )
184 )
185 )
186 )
187 )
188 )
189 )
190 )
191 )
192 )
193 )
194 )
195 )
196 )
197 )
198 )
199 )
200 )
201 )
202 )
203 )
204 )
205 )
206 )
207 )
208 )
209 )
210 )
211 )
212 )
213 )
214 )
215 )
216 )
217 )
218 )
219 )
220 )
221 )
222 )
223 )
224 )
225 )
226 )
227 )
228 )
229 )
230 )
231 )
232 )
233 )
234 )
235 )
236 )
237 )
238 )
239 )
240 )
241 )
242 )
243 )
244 )
245 )
246 )
247 )
248 )
249 )
250 )
251 )
252 )
253 )
254 )
255 )
256 )
257 )
258 )
259 )
260 )
261 )
262 )
263 )
264 )
265 )
266 )
267 )
268 )
269 )
270 )
271 )
272 )
273 )
274 )
275 )
276 )
277 )
278 )
279 )
280 )
281 )
282 )
283 )
284 )
285 )
286 )
287 )
288 )
289 )
290 )
291 )
292 )
293 )
294 )
295 )
296 )
297 )
298 )
299 )
300 )
301 )
302 )
303 )
304 )
305 )
306 )
307 )
308 )
309 )
310 )
311 )
312 )
313 )
314 )
315 )
316 )
317 )
318 )
319 )
320 )
321 )
322 )
323 )
324 )
325 )
326 )
327 )
328 )
329 )
330 )
331 )
332 )
333 )
334 )
335 )
336 )
337 )
338 )
339 )
340 )
341 )
342 )
343 )
344 )
345 )
346 )
347 )
348 )
349 )
350 )
351 )
352 )
353 )
354 )
355 )
356 )
357 )
358 )
359 )
360 )
361 )
362 )
363 )
364 )
365 )
366 )
367 )
368 )
369 )
370 )
371 )
372 )
373 )
374 )
375 )
376 )
377 )
378 )
379 )
380 )
381 )
382 )
383 )
384 )
385 )
386 )
387 )
388 )
389 )
390 )
391 )
392 )
393 )
394 )
395 )
396 )
397 )
398 )
399 )
400 )
401 )
402 )
403 )
404 )
405 )
406 )
407 )
408 )
409 )
410 )
411 )
412 )
413 )
414 )
415 )
416 )
417 )
418 )
419 )
420 )
421 )
422 )
423 )
424 )
425 )
426 )
427 )
428 )
429 )
430 )
431 )
432 )
433 )
434 )
435 )
436 )
437 )
438 )
439 )
440 )
441 )
442 )
443 )
444 )
445 )
446 )
447 )
448 )
449 )
450 )
451 )
452 )
453 )
454 )
455 )
456 )
457 )
458 )
459 )
460 )
461 )
462 )
463 )
464 )
465 )
466 )
467 )
468 )
469 )
470 )
471 )
472 )
473 )
474 )
475 )
476 )
477 )
478 )
479 )
480 )
481 )
482 )
483 )
484 )
485 )
486 )
487 )
488 )
489 )
490 )
491 )
492 )
493 )
494 )
495 )
496 )
497 )
498 )
499 )
500 )
501 )
502 )
503 )
504 )
505 )
506 )
507 )
508 )
509 )
510 )
511 )
512 )
513 )
514 )
515 )
516 )
517 )
518 )
519 )
520 )
521 )
522 )
523 )
524 )
525 )
526 )
527 )
528 )
529 )
530 )
531 )
532 )
533 )
534 )
535 )
536 )
537 )
538 )
539 )
540 )
541 )
542 )
543 )
544 )
545 )
546 )
547 )
548 )
549 )
550 )
551 )
552 )
553 )
554 )
555 )
556 )
557 )
558 )
559 )
560 )
561 )
562 )
563 )
564 )
565 )
566 )
567 )
568 )
569 )
570 )
571 )
572 )
573 )
574 )
575 )
576 )
577 )
578 )
579 )
580 )
581 )
582 )
583 )
584 )
585 )
586 )
587 )
588 )
589 )
590 )
591 )
592 )
593 )
594 )
595 )
596 )
597 )
598 )
599 )
600 )
601 )
602 )
603 )
604 )
605 )
606 )
607 )
608 )
609 )
610 )
611 )
612 )
613 )
614 )
615 )
616 )
617 )
618 )
619 )
620 )
621 )
622 )
623 )
624 )
625 )
626 )
627 )
628 )
629 )
630 )
631 )
632 )
633 )
634 )
635 )
636 )
637 )
638 )
639 )
640 )
641 )
642 )
643 )
644 )
645 )
646 )
647 )
648 )
649 )
650 )
651 )
652 )
653 )
654 )
655 )
656 )
657 )
658 )
659 )
660 )
661 )
662 )
663 )
664 )
665 )
666 )
667 )
668 )
669 )
670 )
671 )
672 )
673 )
674 )
675 )
676 )
677 )
678 )
679 )
680 )
681 )
682 )
683 )
684 )
685 )
686 )
687 )
688 )
689 )
690 )
691 )
692 )
693 )
694 )
695 )
696 )
697 )
698 )
699 )
700 )
701 )
702 )
703 )
704 )
705 )
706 )
707 )
708 )
709 )
710 )
711 )
712 )
713 )
714 )
715 )
716 )
717 )
718 )
719 )
720 )
721 )
722 )
723 )
724 )
725 )
726 )
727 )
728 )
729 )
730 )
731 )
732 )
733 )
734 )
735 )
736 )
737 )
738 )
739 )
740 )
741 )
742 )
743 )
744 )
745 )
746 )
747 )
748 )
749 )
750 )
751 )
752 )
753 )
754 )
755 )
756 )
757 )
758 )
759 )
760 )
761 )
762 )
763 )
764 )
765 )
766 )
767 )
768 )
769 )
770 )
771 )
772 )
773 )
774 )
775 )
776 )
777 )
778 )
779 )
780 )
781 )
782 )
783 )
784 )
785 )
786 )
787 )
788 )
789 )
790 )
791 )
792 )
793 )
794 )
795 )
796 )
797 )
798 )
799 )
800 )
801 )
802 )
803 )
804 )
805 )
806 )
807 )
808 )
809 )
810 )
811 )
812 )
813 )
814 )
815 )
816 )
817 )
818 )
819 )
820 )
821 )
822 )
823 )
824 )
825 )
826 )
827 )
828 )
829 )
830 )
831 )
832 )
833 )
834 )
835 )
836 )
837 )
838 )
839 )
840 )
841 )
842 )
843 )
844 )
845 )
846 )
847 )
848 )
849 )
850 )
851 )
852 )
853 )
854 )
855 )
856 )
857 )
858 )
859 )
860 )
861 )
862 )
863 )
864 )
865 )
866 )
867 )
868 )
869 )
870 )
871 )
872 )
873 )
874 )
875 )
876 )
877 )
878 )
879 )
880 )
881 )
882 )
883 )
884 )
885 )
886 )
887 )
888 )
889 )
890 )
891 )
892 )
893 )
894 )
895 )
896 )
897 )
898 )
899 )
900 )
901 )
902 )
903 )
904 )
905 )
906 )
907 )
908 )
909 )
910 )
911 )
912 )
913 )
914 )
915 )
916 )
917 )
918 )
919 )
920 )
921 )
922 )
923 )
924 )
925 )
926 )
927 )
928 )
929 )
930 )
931 )
932 )
933 )
934 )
935 )
936 )
937 )
938 )
939 )
940 )
941 )
942 )
943 )
944 )
945 )
946 )
947 )
948 )
949 )
950 )
951 )
952 )
953 )
954 )
955 )
956 )
957 )
958 )
959 )
960 )
961 )
962 )
963 )
964 )
965 )
966 )
967 )
968 )
969 )
970 )
971 )
972 )
973 )
974 )
975 )
976 )
977 )
978 )
979 )
980 )
981 )
982 )
983 )
984 )
985 )
986 )
987 )
988 )
989 )
990 )
991 )
992 )
993 )
994 )
995 )
996 )
997 )
998 )
999 )
1000 )
1001 )
1002 )
1003 )
1004 )
1005 )
1006 )
1007 )
1008 )
1009 )
1010 )
1011 )
1012 )
1013 )
1014 )
1015 )
1016 )
1017 )
1018 )
1019 )
1020 )
1021 )
1022 )
1023 )
1024 )
1025 )
1026 )
1027 )
1028 )
1029 )
1030 )
1031 )
1032 )
1033 )
1034 )
1035 )
1036 )
1037 )
1038 )
1039 )
1040 )
1041 )
1042 )
1043 )
1044 )
1045 )
1046 )
1047 )
1048 )
1049 )
1050 )
1051 )
1052 )
1053 )
1054 )
1055 )
1056 )
1057 )
1058 )
1059 )
1060 )
1061 )
1062 )
1063 )
1064 )
1065 )
1066 )
1067 )
1068 )
1069 )
1070 )
1071 )
1072 )
1073 )
1074 )
1075 )
1076 )
1077 )
1078 )
1079 )
1080 )
1081 )
1082 )
1083 )
1084 )
1085 )
1086 )
1087 )
1088 )
1089 )
1090 )
1091 )
1092 )
1093 )
1094 )
1095 )
1096 )
1097 )
1098 )
1099 )
1100 )
1101 )
1102 )
1103 )
1104 )
1105 )
1106 )
1107 )
1108 )
1109 )
1110 )
1111 )
1112 )
1113 )
1114 )
1115 )
1116 )
1117 )
1118 )
1119 )
1120 )
1121 )
1122 )
1123 )
1124 )
1125 )
1126 )
1127 )
1128 )
1129 )
1130 )
1131 )
1132 )
1133 )
1134 )
1135 )
1136 )
1137 )
1138 )
1139 )
1140 )
1141 )
1142 )
1143 )
1144 )
1145 )
1146 )
1147 )
1148 )
1149 )
1150 )
1151 )
1152 )
1153 )
1154 )
1155 )
1156 )
1157 )
1158 )
1159 )
1160 )
1161 )
1162 )
1163 )
1164 )
1165 )
1166 )
1167 )
1168 )
1169 )
1170 )
1171 )
1172 )
1173 )
1174 )
1175 )
1176 )
1177 )
1178 )
1179 )
1180 )
1181 )
1182 )
1183 )
1184 )
1185 )
1186 )
1187 )
1188 )
1189 )
1190 )
1191 )
1192 )
1193 )
1194 )
1195 )
1196 )
1197 )
1198 )
1199 )
1200 )
1201 )
1202 )
1203 )
1204 )
1205 )
1206 )
1207 )
1208 )
1209 )
1210 )
1211 )
1212 )
1213 )
1214 )
1215 )
1216 )
1217 )
1218 )
1219 )
1220 )
1221 )
1222 )
1223 )
1224 )
1225 )
1226 )
1227 )
1228 )
1229 )
1230 )
1231 )
1232 )
1233 )
1234 )
1235 )
1236 )
1237 )
1238 )
1239 )
1240 )
1241 )
1242 )
1243 )
1244 )
1245 )
1246 )
1247 )
1248 )
1249 )
1250 )
1251 )
1252 )
1253 )
1254 )
1255 )
1256 )
1257 )
1258 )
1259 )
1260 )
1261 )
1262 )
1263 )
1264 )
1265 )
1266 )
1267 )
1268 )
1269 )
1270 )
1271 )
1272 )
1273 )
1274 )
1275 )
1276 )
1277 )
1278 )
1279 )
1280 )
1281 )
1282 )
1283 )
1284 )
1285 )
1286 )
1287 )
1288 )
1289 )
1290 )
1291 )
1292 )
1293 )
1294 )
1295 )
1296 )
1297 )
1298 )
1299 )
1300 )
1301 )
1302 )
1303 )
1304 )
1305 )
1306 )
1307 )
1308 )
1309 )
1310 )
1311 )
1312 )
1313 )
1314 )
1315 )
1316 )
1317 )
1318 )
1319 )
1320 )
1321 )
1322 )
1323 )
1324 )
1325 )
1326 )
1327 )
1328 )
1329 )
1330 )
1331 )
1332 )
1333 )
1334 )
1335 )
1336 )
1337 )
1338 )
1339 )
1340 )
1341 )
1342 )
1343 )
1344 )
1345 )
1346 )
1347 )
1348 )
1349 )
1350 )
1351 )
1352 )
1353 )
1354 )
1355 )
1356 )
1357 )
1358 )
1359 )
1360 )
1361 )
1362 )
1363 )
1364 )
1365 )
1366 )
1367 )
1368 )
1369 )
1370 )
1371 )
1372 )
1373 )
1374 )
1375 )
1376 )
1377 )
1378 )
1379 )
1380 )
1381 )
1382 )
1383 )
1384 )
1385 )
1386 )
1387 )
1388 )
1389 )
1390 )
1391 )
1392 )
1393 )
1394 )
1395 )
1396 )
1397 )
1398 )
1399 )
1400 )
1401 )
1402 )
1403 )
1404 )
1405 )
1406 )
1407 )
1408 )
1409 )
1410 )
1411 )
1412 )
1413 )
1414 )
1415 )
1416 )
1417 )
1418 )
1419 )
1420 )
1421 )
1422 )
1423 )
1424 )
1425 )
1426 )
1427 )
1428 )
1429 )
1430 )
1431 )
1432 )
1433 )
1434 )
1435 )
1436 )
1437 )
1438 )
1439 )
1440 )
1441 )
1442 )
1443 )
1444 )
1445 )
1446 )
1447 )
1448 )
1449 )
1450 )
1451 )
1452 )
1453 )
1454 )
1455 )
1456 )
1457 )
1458 )
1459 )
1460 )
1461 )
1462 )
1463 )
1464 )
1465 )
1466 )
1467 )
1468 )
1469 )
1470 )
1471 )
1472 )
1473 )
1474 )
1475 )
1476 )
1477 )
1478 )
1479 )
1480 )
1481 )
1482 )
1483 )
1484 )
1485 )
1486 )
1487 )
1488 )
1489 )
1490 )
1491 )
1492 )
1493 )
1494 )
1495 )
1496 )
1497 )
1498 )
1499 )
1500 )
1501 )
1502 )
1503 )
1504 )
1505 )
1506 )
1507 )
1508 )
1509 )
1510 )
1511 )
1512 )
1513 )
1514 )
1515 )
1516 )
1517 )
1518 )
1519 )
1520 )
1521 )
1522 )
1523 )
1524 )
1525 )
1526 )
1527 )
1528 )
1529 )
1530 )
1531 )
1532 )
1533 )
1534 )
1535 )
1536 )
1537 )
1538 )
1539 )
1540 )
1541 )
1542 )
1543 )
1544 )
1545 )
1546 )
1547 )
1548 )
1549 )
1550 )
1551 )
1552 )
1553 )
1554 )
1555 )
1556 )
1557 )
1558 )
1559 )
1560 )
1561 )
1562 )
1563 )
1564 )
1565 )
1566 )
1567 )
1568 )
1569 )
1570 )
1571 )
1572 )
1573 )
1574 )
1575 )
1576 )
1577 )
1578 )
1579 )
1580 )
1581 )
1582 )
1583 )
1584 )
1585 )
1586 )
1587 )
1588 )
1589 )
1590 )
1591 )
1592 )
1593 )
1594 )
1595 )
1596 )
1597 )
1598 )
1599 )
1600 )
1601 )
1602 )
1603 )
1604 )
1605 )
1606 )
1607 )
1608 )
1609 )
1610 )
1611 )
1612 )
1613 )
1614 )
1615 )
1616 )
1617 )
1618 )
1619 )
1620 )
1621 )
1622 )
1623 )
1624 )
1625 )
1626 )
1627 )
1628 )
1629 )
1630 )
1631 )
1632 )
1633 )
1634 )
1635 )
1636 )
1637 )
1638 )
1639 )
1640 )
1641 )
1642 )
1643 )
1644 )
1645 )
1646 )
1647 )
1648 )
1649 )
1650 )
1651 )
1652 )
1653 )
1654 )
1655 )
1656 )
1657 )
1658 )
1659 )
1660 )
1661 )
1662 )
1663 )
1664 )
1665 )
1666 )
1667 )
1668 )
1669 )
1670 )
1671 )
1672 )
1673 )
1674 )
1675 )
1676 )
1677 )
1678 )
1679 )
1680 )
1681 )
1682 )
1683 )
1684 )
1685 )
1686 )
1687 )
1688 )
1689 )
1690 )
1691 )
1692 )
1693 )
1694 )
1695 )
1696 )
1697 )
1698 )
1699 )
1700 )
1701 )
1702 )
1703 )
1704 )
1705 )
1706 )
1707 )
1708 )
1709 )
1710 )
1711 )
1712 )
1713 )
1714 )
1715 )
1716 )
1717 )
1718 )
1719 )
1720 )
1721 )
1722 )
1723 )
1724 )
1725 )
1726 )
1727 )
1728 )
1729 )
1730 )
1731 )
1732 )
1733 )
1734 )
1735 )
1736 )
1737 )
1738 )
1739 )
1740 )
1741 )
1742 )
1743 )
1744 )
1745 )
1746 )
1747 )
1748 )
1749 )
1750 )
1751 )
1752 )
1753 )
1754 )
1755 )
1756 )
1757 )
1758 )
1759 )
1760 )
1761 )
1762 )
1763 )
1764 )
1765 )
1766 )
1767 )
1768 )
1769 )
1770 )
1771 )
1772 )
1773 )
1774 )
1775 )
1776 )
1777 )
1778 )
1779 )
1780 )
1781 )
1782 )
1783 )
1784 )
1785 )
1786 )
1787 )
1788 )
1789 )
1790 )
1791 )
1792 )
1793 )
1794 )
1795 )
1796 )
1797 )
1798 )
1799 )
1800 )
1801 )
1802 )
1803 )
1804 )
1805 )
1806 )
1807 )
1808 )
1809 )
1810 )
1811 )
1812 )
1813 )
1814 )
1815 )
1816 )
1817 )
1818 )
1819 )
1820 )
1821 )
1822 )
1823 )
1824 )
1825 )
1826 )
1827 )
1828 )
1829 )
1830 )
1831 )
1832 )
1833 )
1834 )
1835 )
1836 )
1837 )
1838 )
1839 )
1840 )
1841 )
1842 )
1843 )
1844 )
1845 )
1846 )
1847 )
1848 )
1849 )
1850 )
1851 )
1852 )
1853 )
1854 )
1855 )
1856 )
1857 )
1858 )
1859 )
1860 )
1861 )
1862 )
1863 )
1864 )
1865 )
1866 )
1867 )
1868 )
1869 )
1870 )
1871 )
1872 )
1873 )
1874 )
1875 )
1876 )
1877 )
1878 )
1879 )
1880 )
1881 )
1882 )
1883 )
1884 )
1885 )
1886 )
1887 )
1888 )
1889 )
1890 )
1891 )
1892 )
1893 )
1894 )
1895 )
1896 )
1897 )
1898 )
1899 )
1900 )
1901 )
1902 )
1903 )
1904 )
1905 )
1906 )
1907 )
1908 )
1909 )
1910 )
1911 )
1912 )
1913 )
1914 )
1915 )
1916 )
1917 )
1918 )
1919 )
1920 )
1921 )
1922 )
1923 )
1924 )
1925 )
1926 )
1927 )
1928 )
1929 )
1930 )
1931 )
1932 )
1933 )
1934 )
1935 )
1936 )
1937 )
1938 )
1939 )
1940 )
1941 )
1942 )
1943 )
1944 )
1945 )
1946 )
1947 )
1948 )
1949 )
1950 )
1951 )
1952 )
1953 )
1954 )
1955 )
1956 )
1957 )
1958 )
1959 )
1960 )
1961 )
1962 )
1963 )
1964 )
1965 )
1966 )
1967 )
1968 )
1969 )
1970 )
1971 )
1972 )
1973 )
1974 )
1975 )
1976 )
1977 )
1978 )
1979 )
1980 )
1981 )
1982 )
1983 )
1984 )
1985 )
1986 )
1987 )
1988 )
1989 )
1990 )
1991 )
1992 )
1993 )
1994 )
1995 )
1996 )
1997 )
1998 )
1999 )
2000 )
2001 )
2002 )
2003 )
2004 )
2005 )
2006 )
2007 )
2008 )
2009 )
2010 )
2011 )
2012 )
2013 )
2014 )
2015 )
2016 )
2017 )
2018 )
2019 )
2020 )
2021 )
2022 )
2023 )
2024 )
2025 )
2026 )
2027 )
2028 )
2029 )
2030 )
2031 )
2032 )
2033 )
2034 )
2035 )
2036 )
2037 )
2038 )
2039 )
2040 )
2041 )
2042 )
2043 )
2044 )
2045 )
2046 )
2047 )
2048 )
2049 )
2050 )
2051 )
2052 )
2053 )
2054 )
2055 )
2056 )
2057 )
2058 )
2059 )
2060 )
2061 )
2062 )
2063 )
2064 )
2065 )
2066 )
2067 )
2068 )
2069 )
2070 )
2071 )
2072 )
2073 )
2074 )
2075 )
2076 )
2077 )
2078 )
2079 )
2080 )
2081 )
2082 )
2083 )
2084 )
2085 )
2086 )
2087 )
2088 )
2089 )
2090 )
2091 )
2092 )
2093 )
2094 )
2095 )
2096 )
2097 )
2098 )
2099 )
2100 )
2101 )
2102 )
2103 )
2104 )
2105 )
2106 )
2107 )
2108 )
2109 )
2110 )
2111 )
2112 )
2113 )
2114 )
2115 )
2116 )
2117 )
2118 )
2119 )
2120 )
2121 )
2122 )
2123 )
2124 )
2125 )
2126 )
2127 )
2128 )
2129 )
2130 )
2131 )
2132 )
2133 )
2134 )
2135 )
2136 )
2137 )
2138 )
2139 )
2140 )
2141 )
2142 )
2143 )
2144 )
2145 )
2146 )
2147 )
2148 )
2149 )
2150 )
2151 )
2152 )
2153 )
2154 )
2155 )
2156 )
2157 )
2158 )
2159 )
2160 )
2161 )
2162 )
2163 )
2164 )
2165 )
2166 )
2167 )
2168 )
2169 )
2170 )
2171 )
2172 )
2173 )
2174 )
2175 )
2176 )
2177 )
2178 )
2179 )
2180 )
2181 )
2182 )
21
```

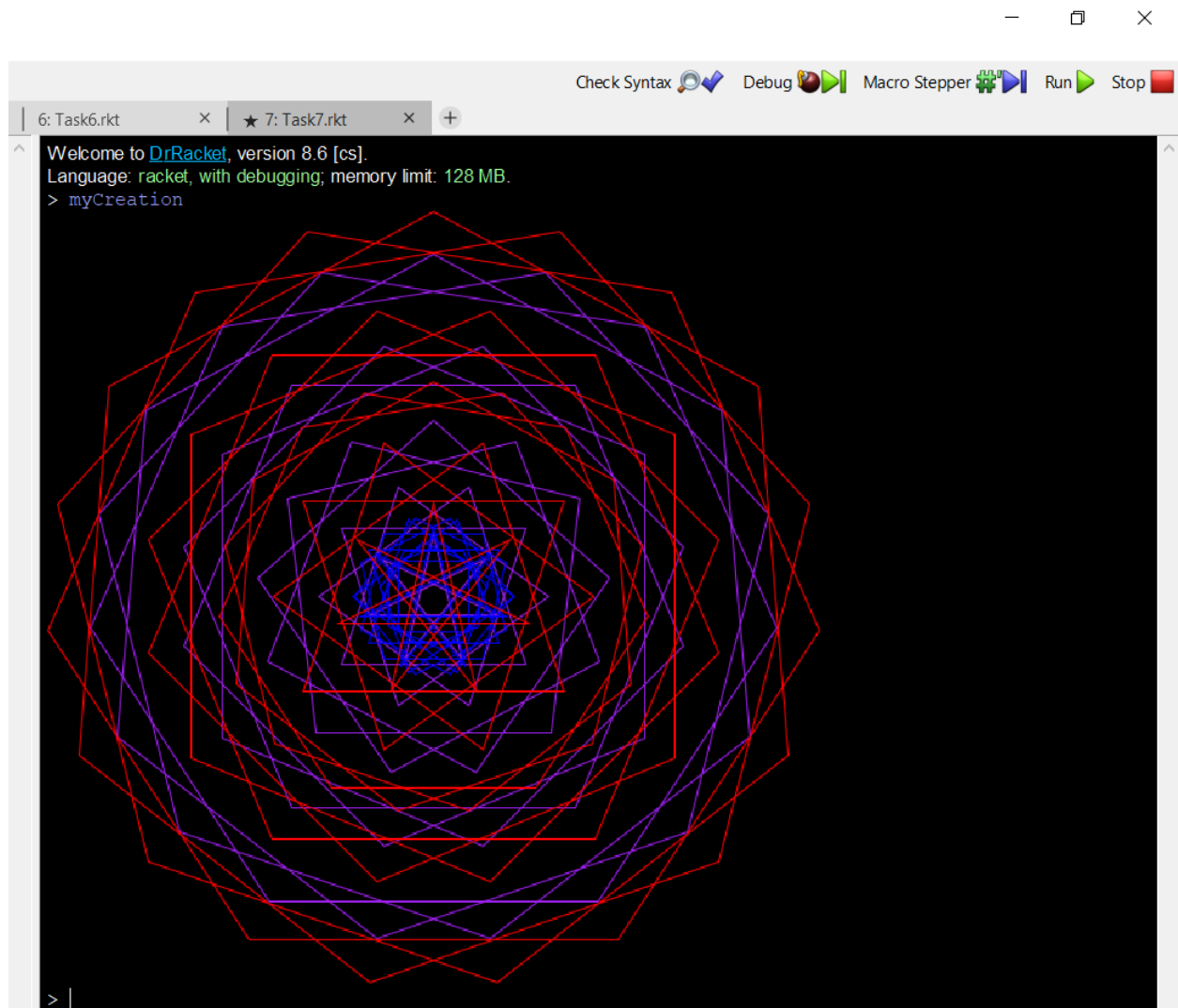
```

45
46 (define basic-tile5 (overlay (pip) basic-tile4 ) )
47
48 (define basic-tile6
49   (overlay/offset (pip) 0 d
50     (overlay/offset (pip) 0 nd
51       (overlay basic-tile4 basic-tile2)
52     )
53   )
54 )
55
56 ;The framed framed tiles
57 ; - Bind one variable to each of the six framed tiles
58
59 (define frame (square side-of-tile "outline" "gray" ) )
60
61 (define tile0 (overlay frame blank-tile) )
62 (define tile1 (overlay frame basic-tile1) )
63 (define tile2 (overlay frame basic-tile2) )
64 (define tile3 (overlay frame basic-tile3) )
65 (define tile4 (overlay frame basic-tile4 ) )
66 (define tile5 (overlay frame basic-tile5 ) )
67 (define tile6 (overlay frame basic-tile6 ) )
68
69 ;Domino generator
70 ; - Function to generate a domino
71
72 (define (domino a b )
73   (beside (tile a) (tile b ) )
74 )
75
76 (define (tile x )
77   (cond
78     ( ( = x 0 ) tile0 )
79     ( ( = x 1 ) tile1 )
80     ( ( = x 2 ) tile2 )
81     ( ( = x 3 ) tile3 )
82     ( ( = x 4 ) tile4 )
83     ( ( = x 5 ) tile5 )
84     ( ( = x 6 ) tile6 )
85   )
86 )

```

Task 7 My Creation:

Demo:



Code:

```
1 | #lang racket
2 | (require 2htdp/image )
3 |
4 | (define myCreation
5 |   (underlay
6 |     (star-polygon 40 7 3 "outline" "purple" )
7 |     (star-polygon 50 10 3 "outline" "purple" )
8 |     (star-polygon 60 13 3 "outline" "purple" )
9 |     (star-polygon 70 16 3 "outline" "purple" )
10 |    (star-polygon 80 19 3 "outline" "purple" )
11 |    (star-polygon 50 4 3 "outline" "blue" )
12 |    (star-polygon 45 7 3 "outline" "blue" )
13 |    (star-polygon 35 10 3 "outline" "blue" )
14 |    (star-polygon 25 14 3 "outline" "blue" )
15 |    (star-polygon 15 16 3 "outline" "blue" )
16 |    (star-polygon 60 7 3 "outline" "red" )
17 |    (star-polygon 70 10 3 "outline" "red" )
18 |    (star-polygon 80 16 3 "outline" "red" )
19 |    (star-polygon 50 19 3 "outline" "red" )
20 |    (star-polygon 90 19 3 "outline" "red" )
21 |   )
22 | )
23 |
24 |
```