CS 241
Data Organization
Memory Management – Fractals

Brooke Chenoweth

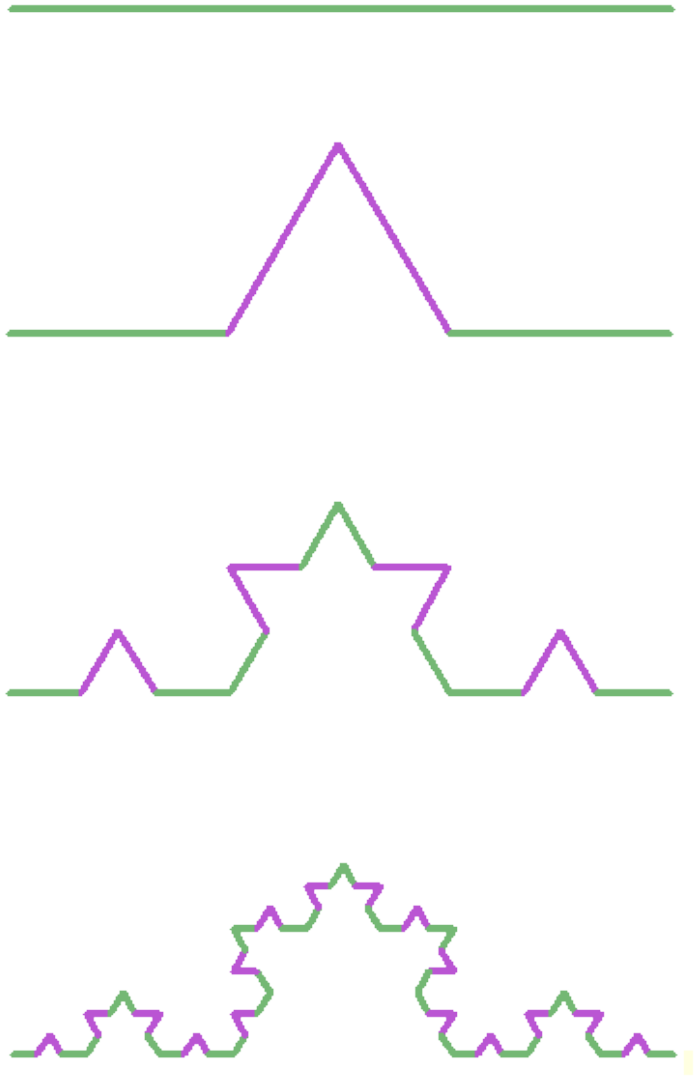
University of New Mexico

Fall 2014

Koch Curve

In its limit, the Koch Curve is:

- Everywhere continuous
- Nowhere Differentiable
- Topological dimension $= 1$
- Box dimension > 1



Koch Curve: A String Representation

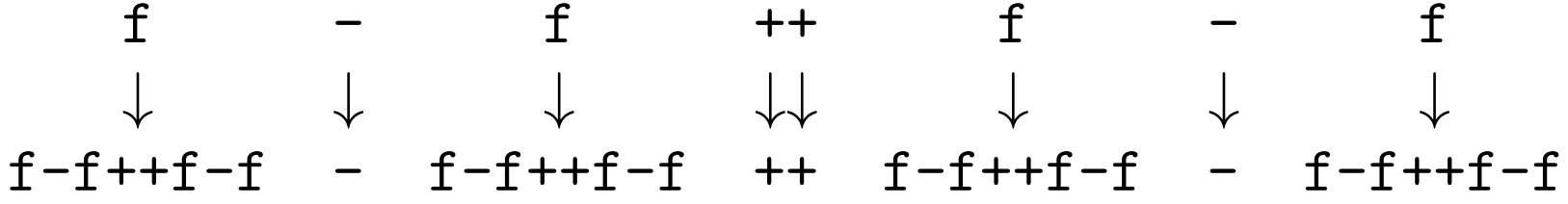
Generation 0

`f`

Generation 1

`f-f++f-f`

Generation 2



- `f`: Draw Unit line Segment.
- `-`: 60° Left Turn
- `+`: 60° Right Turn

substitute() - Part 1 of 2

```
char* substitute(char* source, char c, char* sub)
{
  int i,k;
  int n=0;
  int sourceLen = strlen(source);
  int subLen = strlen(sub);
  char* outStr;
  for (i=0; i<sourceLen; i++)
  {
    if (source[i] == c) n++;
  }
  outStr = malloc(sourceLen + n*(subLen-1));
  /* ... */
  return outStr;
}
```

What is going on here?

substitute() - Part 2 of 2

```
char* substitute(char* source, char c, char* sub)
{
  /* ... */
  outStr = malloc(sourceLen + n*(subLen-1));
  k=0;
  for (i=0; i<sourceLen; i++)
  { if (source[i] == c)
    { strcpy(&outStr[k], sub);
      k += subLen;
    }
    else
    { outStr[k] = source[i];
      k++;
    }
  }
  outStr[k] = '\0';
  return outStr;
}
```

kochCurve()

```
char* kochCurve(int n)
{
  char* baseGen;
  if (n==0)
  {
    baseGen = malloc(2);
    baseGen[0] = 'f';
    baseGen[1] = '\0';
    return baseGen;
  }
  return substitute(kochCurve(n-1),
                    'f', "f-f++f-f");
}
```

Koch curve main()

```
void main(void)
{
   int i;
   for (i = 0; i<4; i++)
   { printf("%s\n", kochCurve(i));
   }
}
```

```
f
f-f++f-f
f-f++f-f-f-f++f-f++f-f++f-f-f-f++f-f
f-f++f-f-f-f++f-f++f-f++f-f-f-f++f-f-f-f++f-f-f-f++f-f-f-f++f-f-
```

valgrind kochCurve

```
HEAP SUMMARY:
    in use at exit: 252 bytes in 10 blocks
  total heap usage: 10 allocs, 0 frees, 252 bytes alloca

LEAK SUMMARY:
   definitely lost: 252 bytes in 10 blocks
   indirectly lost: 0 bytes in 0 blocks
     possibly lost: 0 bytes in 0 blocks
   still reachable: 0 bytes in 0 blocks
        suppressed: 0 bytes in 0 blocks
Rerun with --leak-check=full to see details of leaked me

For counts of detected and suppressed errors, rerun witl
ERROR SUMMARY: 18 errors from 8 contexts (suppressed: 2
```

substitute() - Where to call free?

```
1  char* substitute(char* source, char c, char* sub)
2  {
3    /* ... */
4    char* outStr = malloc(sourceLen + n*(subLen-1));
5    int k=0;
6    for (i=0; i<sourceLen; i++)
7    { if (source[i] == c)
8      { sprintf(&outStr[k], sub);
9        k += subLen;
10     }
11     else
12     { outStr[k] = source[i];
13       k++;
14     }
15   }
16   outStr[k] = '\0';
17   free(source);
18   return outStr;
```

Are *all parts of the program* done with source at line 17?

valgrind kochCurve_v2

```
HEAP SUMMARY:
    in use at exit: 194 bytes in 4 blocks
  total heap usage: 10 allocs, 6 frees, 252 bytes alloca

LEAK SUMMARY:
   definitely lost: 194 bytes in 4 blocks
   indirectly lost: 0 bytes in 0 blocks
     possibly lost: 0 bytes in 0 blocks
   still reachable: 0 bytes in 0 blocks
        suppressed: 0 bytes in 0 blocks
Rerun with --leak-check=full to see details of leaked me

For counts of detected and suppressed errors, rerun witl
ERROR SUMMARY: 18 errors from 8 contexts (suppressed: 2
```

Koch curve main() Leak here?

```
void main(void)
{
  int i;
  for (i = 0; i<4; i++)
  {
    printf("%s\n", kochCurve(i));
  }
}
```

```
void main(void)
{
  int i;
  char* curve;
  for (i = 0; i<4; i++)
  {
    curve = kochCurve(i);
    printf("%s\n", curve);
    free(curve);
  }
}
```

valgrind kochCurve_v3

```
HEAP SUMMARY:
    in use at exit: 0 bytes in 0 blocks
  total heap usage: 10 allocs, 10 frees, 252 b

All heap blocks were freed -- no leaks are pos

For counts of detected and suppressed errors,
ERROR SUMMARY: 18 errors from 8 contexts (supp
```

Still have some errors. Hmm...

substitute() – Where's the bug?

```
1  char* substitute(char* source, char c, char* sub)
2  { /* ... */
3    char* outStr = malloc(sourceLen + n*(subLen-1));
4    int k=0;
5    for (i=0; i<sourceLen; i++)
6    { if (source[i] == c)
7      { sprintf(&outStr[k], sub);
8        k += subLen;
9      }
10     else
11     { outStr[k] = source[i];
12       k++;
13     }
14   }
15   outStr[k] = '\0';
16   free(source);
17   return outStr;
18 }
```

What is wrong with line 3?

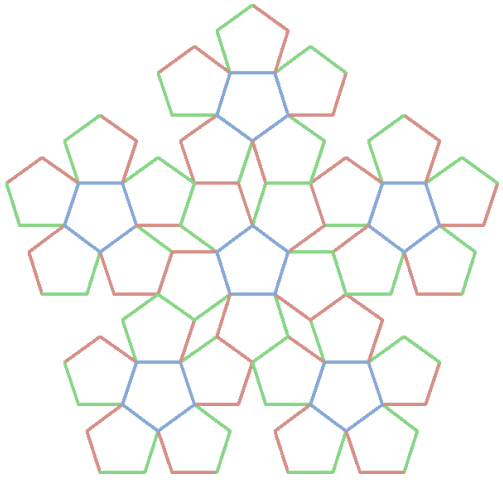
valgrind kochCurve_v4

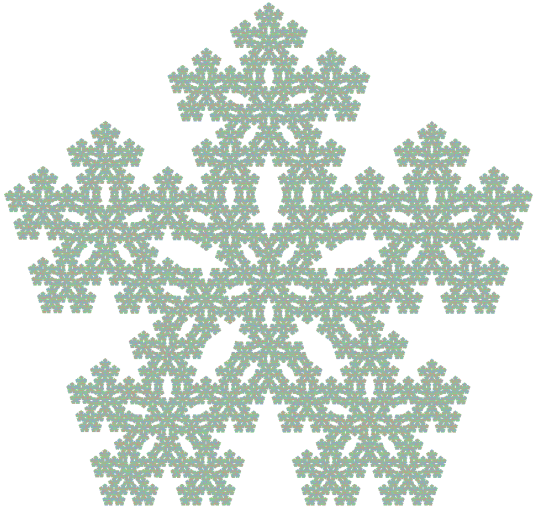
```
HEAP SUMMARY:
    in use at exit: 0 bytes in 0 blocks
  total heap usage: 10 allocs, 10 frees, 258 b

All heap blocks were freed -- no leaks are pos

For counts of detected and suppressed errors,
ERROR SUMMARY: 0 errors from 0 contexts (suppr
```

Penrose Snowflake





- `f`: Draw Unit line Segment.
- `-`: 36° Left Turn
- `+`: 36° Right Turn
- Axiom: `f--f--f--f--f`
- Replacement Rule: `f` $\rightarrow$ `f--f--f-----f+f--f`