Recursion is a programming technique that involves a method calling itself. Recursion is possible because of the way frames and the stack work. Here is how.

Recursive exponential calculation

```
int exp(int i)
{
  int total;
  if (i==1)
    total = 1;
  else
  {
    total = (i * exp(i-1));
  }
  return total;
}
```

- To calculate exponential
- 5! = 5*4*3*2*1
- 4! = 4*3*2*1
- 5! = 5 * 4!
- 1! = 1

 recursion

```
</div>
```

Program starts

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { total = (i * exp(i-1));
} return total; }

- main() frame created as usual
- myExp created as a variable in main() frame

 recursion

```
</div>
```

exp(5) Called

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { total = (i * exp(i-1));
} return total; }

- exp(5) frame created
- local variable i created for the parameter
- parameter copied in to local variable i

 recursion

```
</div>
```

Local Variable total created in exp(5) frame

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { total = (i * exp(i-1));
} return total; }

- exp(5) frame created
- local variable i created for the parameter
- parameter copied in to local variable i

 recursion

```
</div>
```

i > 1 so exp(4) Called

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { ***total = (i * exp(i-1));*** } return total; }

- i is not 1, total = 5 * exp(4)
- call exp(4)
- exp(4) frame added to stack
- parameter value, 4, copied to local variable i in exp(4) frame
- total created exp(4) frame

 recursion

```
</div>
```

i > 1 so exp(3) Called

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { ***total = (i * exp(i-1));*** } return total; }

- i is not 1, total = 4 * exp(3)
- call exp(3)
- exp(3) frame added to stack
- parameter value, 3, copied to local variable i in exp(3) frame
- total created exp(3) frame

 recursion

```
</div>
```

i > 1 so exp(2) Called

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { ***total = (i * exp(i-1));*** } return total; }

- i is not 1, total = 3 * exp(2)
- call exp(4)
- exp(2) frame added to stack
- parameter value, 2, copied to local variable i in exp(2) frame
- total created exp(2) frame

 recursion

```
</div>
```

i > 1 so exp(1) Called

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { ***total = (i * exp(i-1));*** } return total; }

- i is not 1, total = 2 * exp(1)
- call exp(1)
- exp(1) frame added to stack
- parameter value, 1, copied to local variable i in exp(1) frame
- total created exp(1) frame

 recursion

```
</div>
```

i == 1 We stop recursing

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { ***total = (i * exp(i-1));*** } ***return total;*** }

- i is 1, total = 1
- return 1 to calling method - exp(2)

 recursion

```
</div>
```

We finalise exp(2)

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { ***total = (i * exp(i-1));*** } ***return total;*** }

- total = 2 * value returned here = 2 * 1 = 2
- return 2 to calling method - exp(3)

 recursion

```
</div>
```

We finalise exp(3)

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { ***total = (i * exp(i-1));*** } ***return total;*** }

- total = 3 * value returned here = 3 * 2 = 6
- return 6 to calling method - exp(4)

 recursion

```
</div>
```

We finalise exp(4)

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { ***total = (i * exp(i-1));*** } ***return total;*** }

- total = 4 * value returned here = 4 * 6 = 24
- return 24 to calling method - exp(5)

 recursion

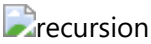
```
</div>
```

We finalise exp(5)

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { ***total = (i * exp(i-1));*** } ***return total;*** }

- total = 5 * value returned here = 5 * 24 = 120
- return 120 to calling method - main()

 recursion

```
</div>
```

Program ends, stack is empty

```
main()
{
  int myExp = exp(5);
}
```

int exp(int i) {
int total; if (i==1) total = 1; else { total = (i * exp(i-1)); } return total; }

- program ends
- main() frame deleted

 recursion

```
</div>
```