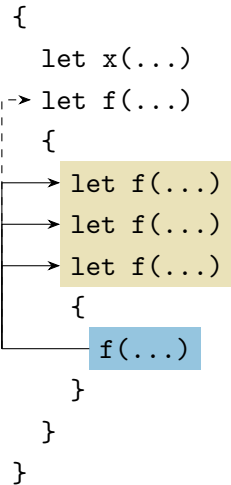
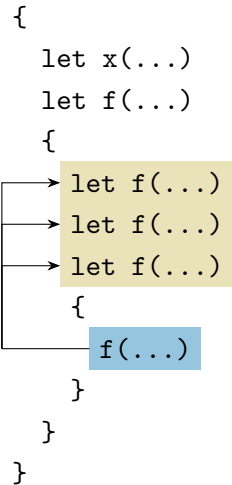
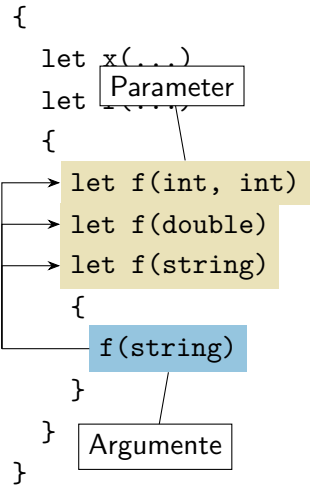


```
{  
  let x(...)  
  let f(...)  
  {  
    let f(...)  
    let f(...)  
    let f(...)  
    {  
      f(...)  
    }  
  }  
}
```

```
{  
  let x(...)  
→ let f(...)  
  {  
    let f(...)  
    let f(...)  
    let f(...)  
    {  
      f(...)  
    }  
  }  
}
```







```
{  
  let x(...)  
  let f(...)  
  {  
    let f(int, int)  
    let f(double)  
    let f(string)  
    {  
      f(string)  
    }  
  }  
}
```

The diagram illustrates function calls within a nested block structure. A vertical line on the left side of the code has two arrows pointing to the right. The top arrow points to the line `let f(double)`, and the bottom arrow points to the line `f(string)`. These two lines are highlighted with a light yellow background. The line `f(string)` is also highlighted with a light blue background.

```
{  
  let x(...)  
  let f(...)  
  {  
    let f(int, int)  
    let f(double)  
    let f(string)  
    {  
      f(string)  
    }  
  }  
}
```

The diagram illustrates the resolution of a function call `f(string)` within a nested scope. A line originates from the `f(string)` call in the innermost block and points to the `let f(string)` definition in the same block, indicating that the call is resolved to the local function definition.