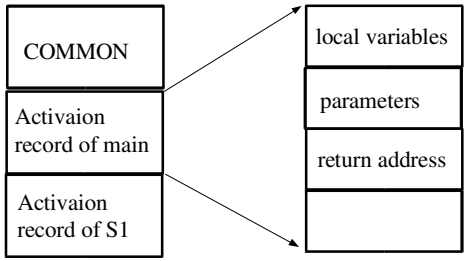
Lecture 22

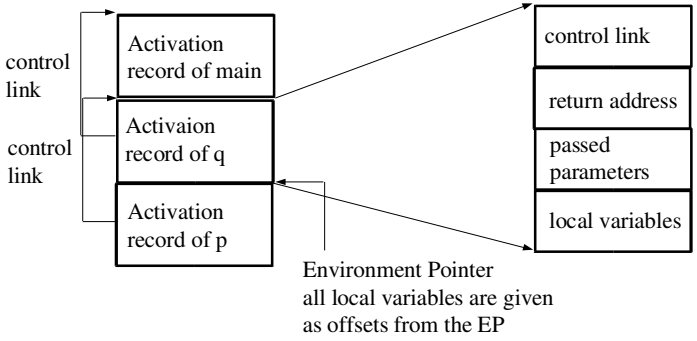
- Leftovers from last lecture
- Object-oriented programming
  - Louden Chapters 10

Procedure Environments, Activations and Allocation



Fortran static evnironment no recursion or nested procedures

Stack-Based Runtime



Nested procedures

- C and Fortran no nested procedures
  - all non-local references global (easy to find)
- Nested procedures (Pascal, Ada, Modula-2)
- Following the control links results in dynamic scoping rather than lexical
- Additional field called access link : link to lexical or defining environment
- closure <ep, ip>

Maximum flexibility

- Procedures are first class values: they can be created at will
- Stack-based environment impossible
- Basically have to store full environment and code (closure) for everything (ML, Scheme, LISP)
- Automatic reclamation of storage
  - Reference counting
  - Garbage collection

Dynamic Memory Management

- Never deallocate
  - Large memory requirement
- Maintaing free space
  - list of free blocks – coalescing, fragmentation (done with disk drives too)
- Reference counting (eager)
- Mark and sweep
- Generational garbage collection

Mark and sweep

- Lazy: Allocator runs out of space
- First pass: Follow all pointers recursively and mark everything reachable (extra bit)
- Second pass: Move all unreferenced cells back to free list
- Problem: processing delays

Generational garbage collection

- Spread cost more evenly
- Allocated objects that survive long enough are simply copied to permanent space and never get reallocated
- Only newer storage allocations need to be searched

Object-oriented programming

- Three major characteristics
  - Data Hiding
  - Inheritance
  - Dynamic Binding

History

- Simula 1967
- Simulation of the real world – objects
- Software reuse
  - Extension
  - Restriction
  - Redefinition
  - Abstraction
  - Polymorphization

Data Hiding

- Global variables can pottentially be accesed and updated by every part of the program
- 1970 David Parnas – Information Hiding
  - module
- ML module system, Ada packages, Modula-2 modules, C++ namespaces, Java packages are examples of information hiding

Inheritance

- Object class is a set of objects that share some operations
- Objects are first-class values (user-defined types)
- Inheritance = ability to organize object classes into an hierarchy of bases classes and derived classes (super and sub)

Dynamic Binding

- Way to distinguish between the general properties of a "base" class and the properties of a "specific kind of" derived class
- The classic example
  - Shape
    - center, color, draw
  - Circle, Triangle, Square

The classic solution

```
enum Kind {circle, triangle, square};

class Shape{
    Kind k;     // type field
    Point center;
    Color color;

public:
    void draw();
    void rotate(int);
}
```

```
void Shape::draw()
{
    switch(k) {
    case circle:
        // draw circle
        break;
    case triangle:
        // draw triangle
        break;
....
}

// PROBLEM: must change
for every new shape
```


The object-oriented solution(C++)

```
class Shape {
    Point center;
    Color col;

public:
    Point where() {return center;}
    virtual void draw() = 0;
    virtual void rotate(int angle) = 0;
}
```

Now the code works for any kind of shape. The appropriate rotate functions is called at run-time (dynamic binding)

```
void rotate_all(vector<Shape *>& v, int angle)
{ for (int i=0; i < v.size(); ++i)
        v[i]->rotate(angle);
}
```


Smalltalk

- Pure-object oriented language
  - everything is an object
- Xerox Palo Alto Research Center 1970s
  - other contributions: mouse, windows, ide
- Self-contained interactive programming system (still a revolutionary idea)
- Alan Key – The best way to predict the future is to invent it – Lot's of little computers (objects)

Primitive Objects

- Integer, Float, Boolean are objects
- expression m+n requests m to return the sum of its own value and that of integer object n; the result is another Integer object
- Dynamic Typing
  - Picture can contain arbitrary objects
  - Flexibility but insecurity

Smalltalk notation

Eo is evaluated to determine the receiver object:

"Eo I" perform operation I, with no arguments. Example: n squarred
"Eo * E1" requests the receiver object to performed the operator named * with the object yielded by E1 as argument. Example: n * 4
"Eo I1:E1....In:En" requests the receiver object to perform the operation named 'I1:....In', with the objects yielded by E1..En as arguments. Example: 'm between: 1 and: n-1' where the operation name is 'between: and:'

For example an array:
monthsize at: 2 put: 29

Assignment expression:
V <- E
n <- n – m

A smalltalk example

```
placeat: xnew and: ynew
    x <- xnew . y <- ynew                    (. is the sequencing operator)
xcoord
    ^ x
ycoord
    ^ y

moveby: xshilf and: yshift
    x <- x + xshift.    y<- y + yshift

distance: other
    ^ ( (x – other xcoord) squarred
      + (y – other ycoord) squared) sqrt
```


History C++

- Started as "C with classes" by B.Stroustrup
  - AT&T Bell Labs 1980s
- Influenced by Simula67
- Based on C (easy portability, interoperation with traditional C environments (Unix))
- One of the most widely used languages
- VERY COMPLEX and BIG but can be learned in smaller chunks

Motivation

- Systems programming
  - No runtime overhead
  - Explicit memory management
- Support
  - Modularity – Information Hiding
  - Inheritance
  - Dynamic Binding (controlled)
  - Generic programming (templates)