The Graph Programming Language

Ephraim Park, Peiqian Li, Qingxiang Jia

Overview - Basics

- Control Structures
    - if, for, while (C-like syntax)
- Variable Scope
    - Local (inside a function)
    - Global
- Entry point
    - Main, void main (Bond, James Bond)
- Data types
    - int
    - char
    - string
    - ...

Overview - Highlights

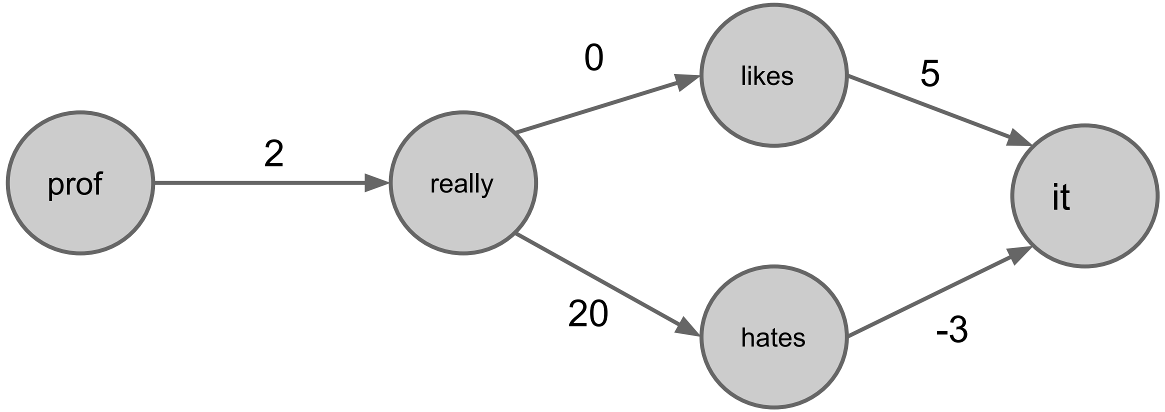
- Support user defined functions
  - ret_typ func_name(para1 … paraN){}
- Support intuitive graph declaration
- Support multi-dimensional array
- All non-primitive vars passed by reference (same as Java)

Overview - Tutorial

```
String[] arrStr = someFuncRetArrStr();

String[3] arrStr1;
arrStr1[0] = “presentation”;

int n;
n = 4;

int m = 5;
```

Overview - Tutorial

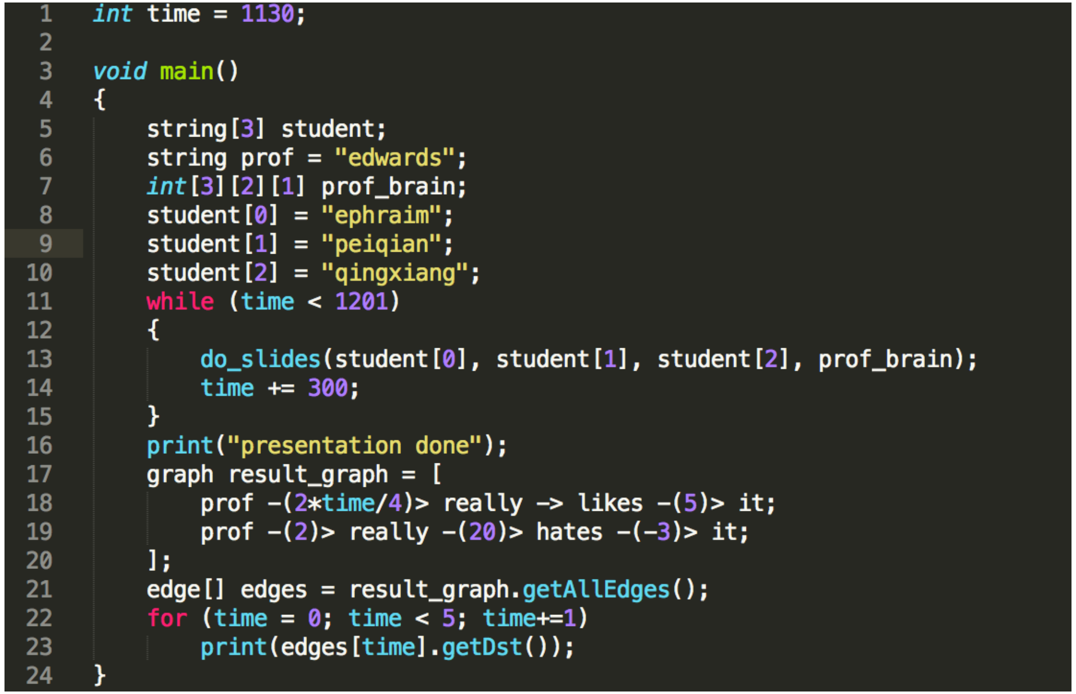
```
graph result_graph = [
        prof -(2*time/4)> really -> likes -(5)> it;
        prof -(2)> really -(20)> hates -(-3)> it;
];
```



Overview - Tutorial

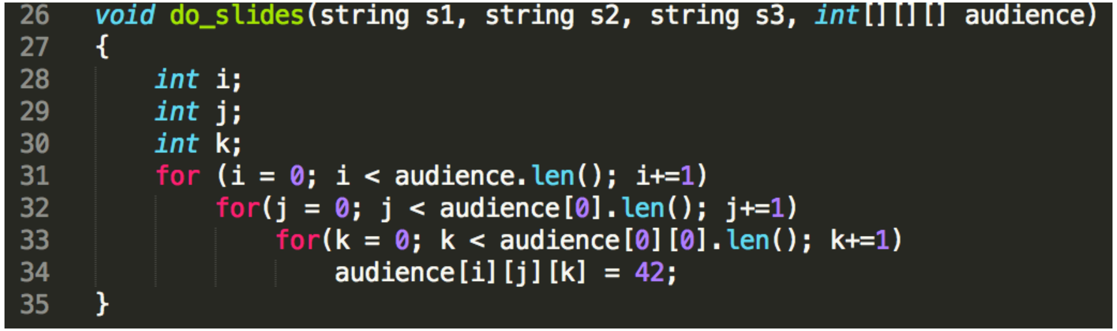
```
while (time < 1201)
{
    do_slides(student[0], student[1], student[2], prof_brain);
    time += 300;
}



for (i = 0; i < audience.len(); i+=1)
        for(j = 0; j < audience[0].len(); j+=1)
            for(k = 0; k < audience[0][0].len(); k+=1)
                audience[i][j][k] = 42;
```

Overview - Code Gathering



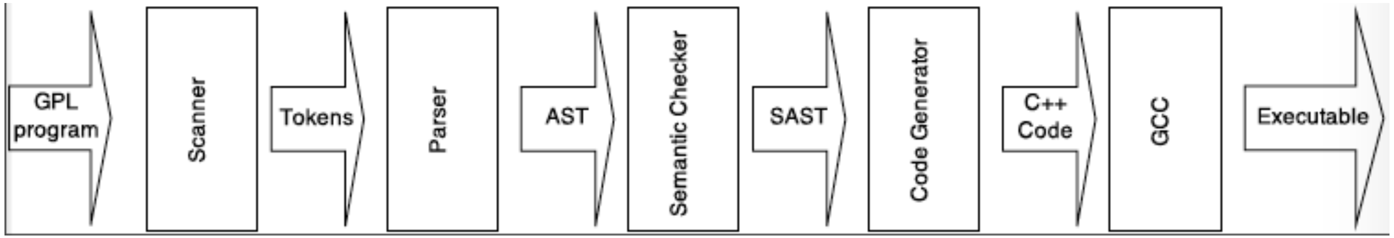
```
int time = 1130;

void main()
{
    string[3] student;
    string prof = "edwards";
    int[3][2][1] prof_brain;
    student[0] = "ephraim";
    student[1] = "peiqian";
    student[2] = "qingxiang";
    while (time < 1201)
    {
        do_slides(student[0], student[1], student[2], prof_brain);
        time += 300;
    }
    print("presentation done");
    graph result_graph = [
        prof -(2*time/4)> really -> likes -(5)> it;
        prof -(2)> really -(20)> hates -(-3)> it;
    ];
    edge[] edges = result_graph.getAllEdges();
    for (time = 0; time < 5; time+=1)
        print(edges[time].getDst());
}
```

Overview - Code Gathering



```
void do_slides(string s1, string s2, string s3, int[][][] audience)
{
    int i;
    int j;
    int k;
    for (i = 0; i < audience.len(); i+=1)
        for(j = 0; j < audience[0].len(); j+=1)
            for(k = 0; k < audience[0][0].len(); k+=1)
                audience[i][j][k] = 42;
}
```

Architecture



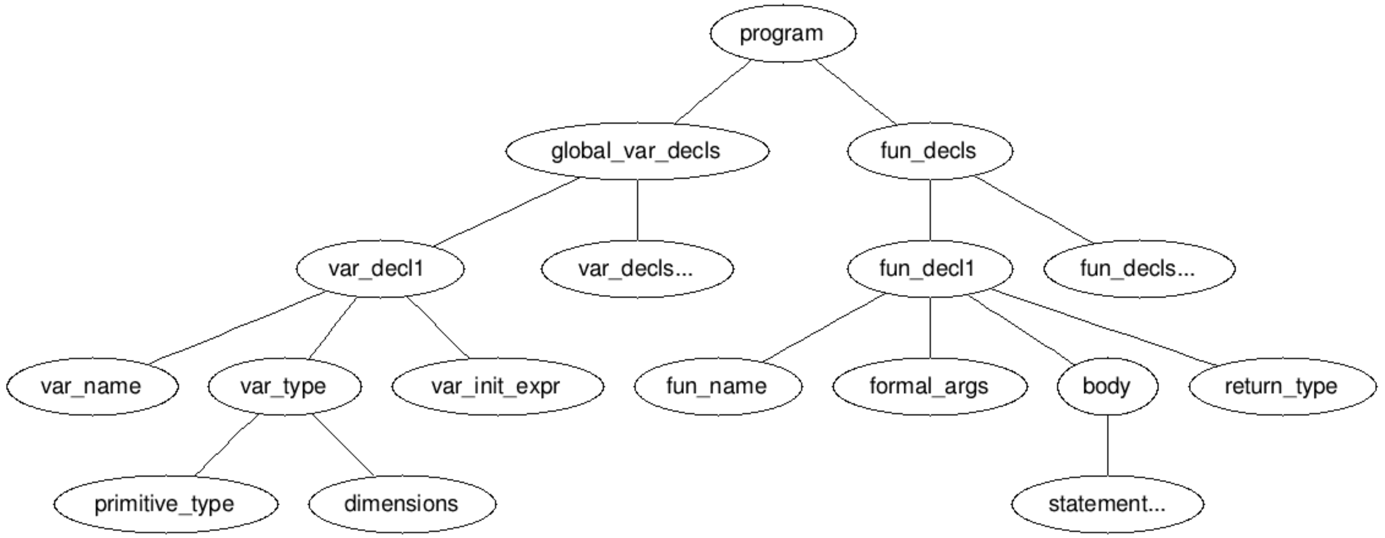
Parser

- Method calls are translated into regular function calls
    - ex) a.sort() → sort(a)
- Graph Literal is a list of edge tuples (src, dest, weight)
- Every variable is an array
    - ex) int a; // a is a zero dimensional array

Semantic Checker

- Type Check
- Variable and Function reference check (Environment)
    - v_context kept information about variables
        - local variable declaration is just a statement and can be done in the middle of the function body
        - StringMap that maps variable name to its type and declaration level
    - f_context kept information about functions
        - StringMap that maps function name to list of function information (parameter and return type)

Abstract Syntax Tree Structure



Code Generation - Array

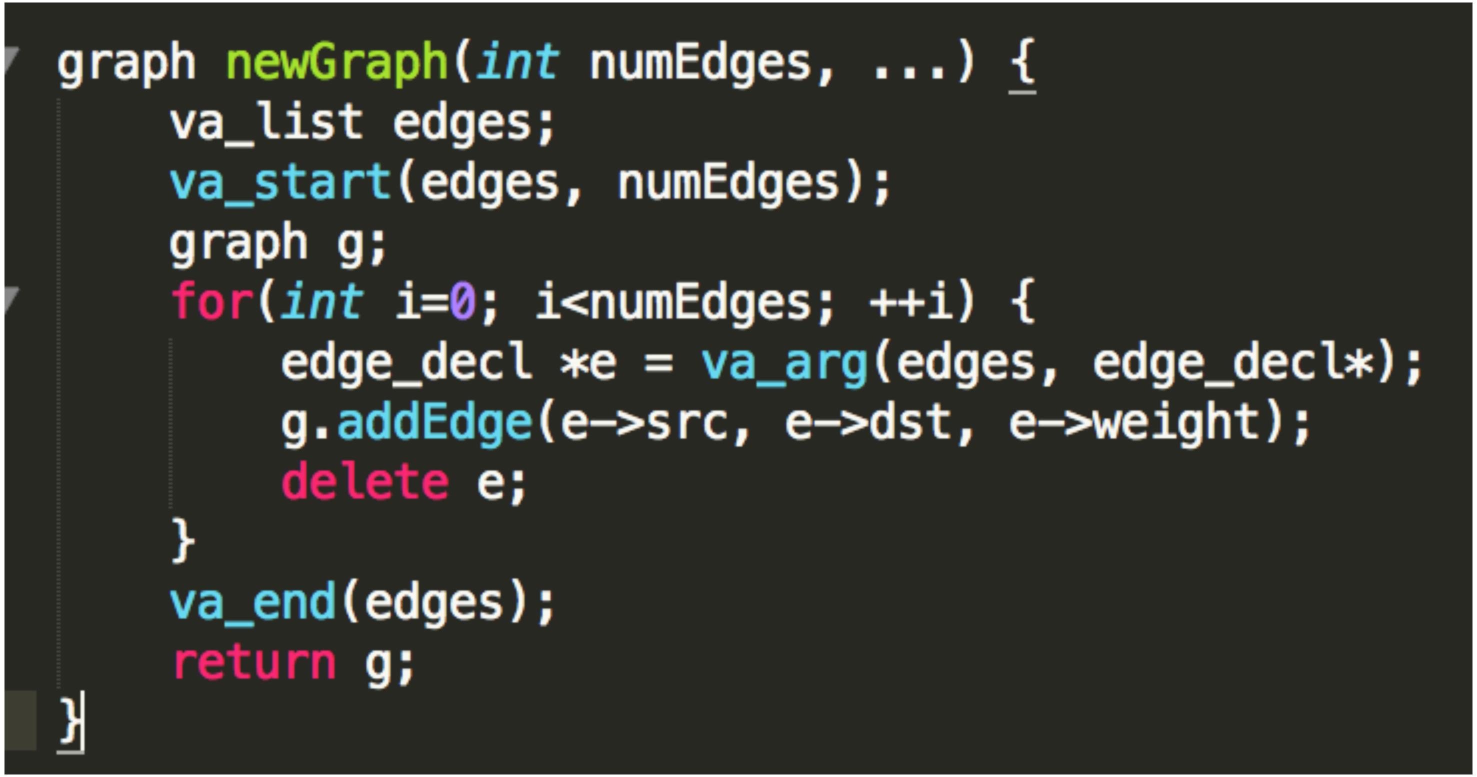
- GPL: string[4][2][8] a;
- C++: vector<vector<vector<string>>> a;

```
a.resize(4);
for(int i=0; i<4; ++i) a[i].resize(2);
for(int i=0; i<4; ++i)
    for(int j=0; j<2; ++j)
        a[i][j].resize(8);
```

Code Generation - Graph

- GPL:
```
void foo(graph g, int t) { … }
        void main() { foo([ a-(5)>b; ], 6); }
```
- C++:
```
void foo(const graph &_g, int t) {
            graph &g = (graph &)_g;
            …
    }
    int main() {
            foo( newGraph(new edge(a, b, 5)), 6 );
            return 0;
    }
```

NewGraph()



```
graph newGraph(int numEdges, ...) {
    va_list edges;
    va_start(edges, numEdges);
    graph g;
    for(int i=0; i<numEdges; ++i) {
        edge_decl *e = va_arg(edges, edge_decl*);
        g.addEdge(e->src, e->dst, e->weight);
        delete e;
    }
    va_end(edges);
    return g;
}
```

Lesson Learned

Ephraim Park

- Really think through the language before start coding
- Whenever making a design decision think about how that decision will be represented in target code
- Try to learn Ocaml in the beginning of the semester!

Peiqian Li

- Really try to learn Ocaml as early as possible!
- When the code doesn't work, in addition to starring at it blankly, you can print stuff out ("ignore (print_endline xxx)"), and/or turning on backtrace and verbose parsing

  (export OCAMLRUNPARAM=b or p).

Qingxiang Jia

- We need comprehensive test cases.