Compiler Construction

Lecture # 05

Mr. Usman Wajid

usman.wajid@nu.edu.pk

February 7, 2023

Symbol Table Management

- An essential function of a compiler to record variable names and collect information about various attribute corresponding to each name

Symbol Table Management

- An essential function of a compiler to record variable names and collect information about various attribute corresponding to each name
- for variable names, the attributes may provide information about the storage allocated for each name, its type, and its scope

Symbol Table Management

- An essential function of a compiler to record variable names and collect information about various attribute corresponding to each name
- for variable names, the attributes may provide information about the storage allocated for each name, its type, and its scope
- for function (or procedure names), the attributes may provide the number, type and sequence of arguments, the method of passing each argument and the return type

Symbol Table Management

- An essential function of a compiler to record variable names and collect information about various attribute corresponding to each name
- for variable names, the attributes may provide information about the storage allocated for each name, its type, and its scope
- for function (or procedure names), the attributes may provide the number, type and sequence of arguments, the method of passing each argument and the return type
- hence, the symbol table is a data structure containing a record for each variable name with corresponding fields of attributes

Symbol Table Management

- An essential function of a compiler to record variable names and collect information about various attribute corresponding to each name
- for variable names, the attributes may provide information about the storage allocated for each name, its type, and its scope
- for function (or procedure names), the attributes may provide the number, type and sequence of arguments, the method of passing each argument and the return type
- hence, the symbol table is a data structure containing a record for each variable name with corresponding fields of attributes
- it should allow the compiler to find, store and retrieve data from the record quickly

Compiler Construction Tools

Some of specialized tools use in compiler construction are as follows,

1. **Parser Generators:** they automatically produce syntax analyzers from grammatical description of a programming language

Compiler Construction Tools

Some of specialized tools use in compiler construction are as follows,

1. **Parser Generators:** they automatically produce syntax analyzers from grammatical description of a programming language
2. **Scanner Generators:** they produce stream of tokens based on regular expression description

Compiler Construction Tools

Some of specialized tools use in compiler construction are as follows,

1. **Parser Generators:** they automatically produce syntax analyzers from grammatical description of a programming language
2. **Scanner Generators:** they produce stream of tokens based on regular expression description
3. **Syntax-directed Translation Engines:** they produce collection of routines for walking a parse tree and generating intermediate code

Compiler Construction Tools

Some of specialized tools use in compiler construction are as follows,

1. **Parser Generators:** they automatically produce syntax analyzers from grammatical description of a programming language
2. **Scanner Generators:** they produce stream of tokens based on regular expression description
3. **Syntax-directed Translation Engines:** they produce collection of routines for walking a parse tree and generating intermediate code
4. **Code-generator Generators:** they produce target machine code by manipulating each rule for translating each operation of the intermediate code

Compiler Construction Tools

5. **Data-flow Analysis Engines:** facilitates the collection of information regarding the transmission flow of values from one part of program to each other part. It is a key part of code optimization

Compiler Construction Tools

5. **Data-flow Analysis Engines:** facilitates the collection of information regarding the transmission flow of values from one part of program to each other part. It is a key part of code optimization
6. **Compiler-construction Toolkits:** they provide an integrated set of routines for construction various phases of a compiler.

Assignment # 01

1. The Evolution of Programming Languages and its Impact on Computer Performance
2. Programming Language Generations
3. its Impact on Computer Performance

Compiler Applications

1. Implementation of high level languages

Compiler Applications

1. Implementation of high level languages
2. optimization for computer architectures

Compiler Applications

1. Implementation of high level languages
2. optimization for computer architectures
3. designing in new computer architectures

Compiler Applications

1. Implementation of high level languages
2. optimization for computer architectures
3. designing in new computer architectures
4. program translations

Compiler Applications

1. Implementation of high level languages
2. optimization for computer architectures
3. designing in new computer architectures
4. program translations
5. software productivity tools

Programming Language Basics

Static Policy

A policy that allows the compiler to address an issue then it is known as static policy

Dynamic Policy

A policy that addresses an issue during execution or run-time is known as dynamic policy

Programming Language Basics continued . . .

Environment

It refers to the mapping of names (variables) to locations (memory addresses).

Environments may change according to the scope rules of a language

States

It refers to the mapping of locations (addresses) to values

Programming Language Basics continued . . .

Identifier

It is a string of characters (letters or digits), that refers to an entity, such as a data object, a procedure (function), a class or a type

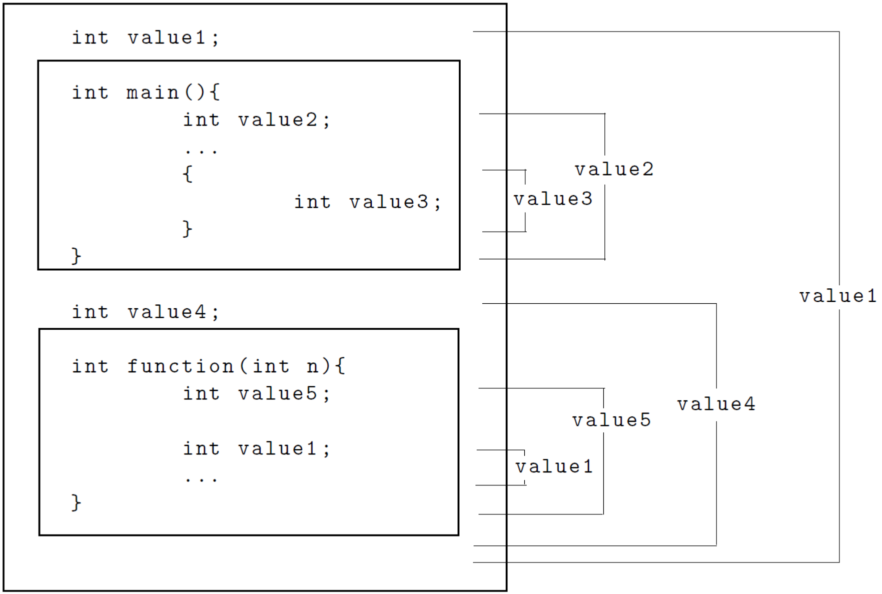
e.g., int result, class Box , void add, struct node etc

Variables

A variable is an instance of an identifier that refers to a particular location in memory.

An identifier can be declared more than once, each such `declaration` introduces a new variable

Variable Scope



```
int value1;

int main(){
        int value2;
        ...
        {
                int value3;
        }
}

int value4;

int function(int n){
        int value5;

        int value1;
        ...
}
```

Programming Language Basics continued . . .

Structure

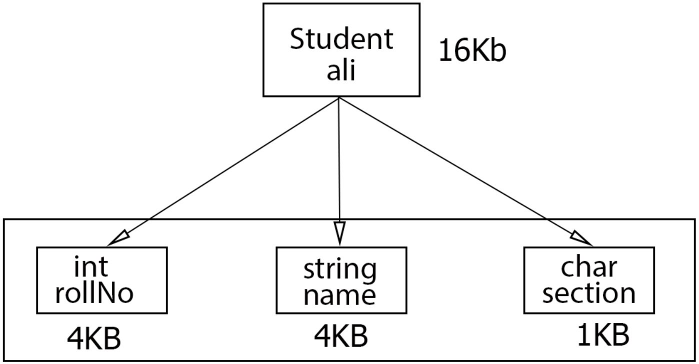
A structure is a collection of related data items that can be referenced with a single name

- The data items in structure are called members
- Unlike arrays, a structure can store members of different types
- Example code,

```
struct Student{
        int rollNo;
        string name;
        char section;
};

Student ali;
```

Structures


Student
ali
16Kb
int
rollNo
4KB
string
name
4KB
char
section
1KB

Programming Language Basics continued . . .

Function

It groups a number of statements into a single unit and assigns it a name

- It can be then invoked from other parts of program
- The function's body is stored in only one place of the memory
- Example,

```
void display(){
            ...
            ...
            ...
}
```