CS335A Specs

Compiler Design
Indian Institute of Technology, Kanpur

Group-4 :
Akhil Hooda (190087)
Aman Jaiswal (190106)
Dinesh Ram (190304)
Shubham Kumawat (190833)
Prof.: Dr. Amey Karkare, Dr. Subhajit Roy

Specs

Source language : C++
Implementation language : C
Target language : MIPS

Basic Features :

- **Native Data types**
  - **Integer** : An integer is a sequence of digits 0-9 which is not a fraction.
    Keywords :
    * int - 4 bytes (-2,147,483,648 to 2,147,483,647)
    * long int - 4 bytes (-2,147,483,648 to 2,147,483,647)
    * unsigned int - 4 bytes (0 to 4,294,967,295)
    * unsigned long - 8 bytes (0 to 4,294,967,295)
    * short int - 2 bytes (-32,768 to 32,767)
    * unsigned short int - 2 bytes (0 to 65,535)
  - **Double, Float** : Double,Float are real numbers.
    Keywords :
    * double - 8 bytes
    * float - 4 bytes
  - **Character** : It is collection of various symbols,letters,digits etc.
    Keywords :
    * char - 1 byte (-128 to 127)
    * unsigned char - 1 byte (0 to 255)

- **Strings** : A string is a collection of characters.
  Keywords :
  * string
- **Boolean** : Boolean are conditional statements which have return type of true or false (1 or 0).
  Keywords :
  * bool - 1 byte (two values: True(1) and False(0))

- **Operators**
  - **Arithmetic**
    Keywords :
    + , - , / , *
  - **Assignment**
    Keywords :
    = , += , -= , /= , *= , ∧=
  - **Increment and Decrement**
    Keywords :
    x++ , ++x , x- - , - -x
  - **Comparison**
    Keywords :
    == , != , >= , <= , > , <
  - **Logical**
    Keywords :
    && , || , !
  - **Bit Shifting**
    Keywords :
    >> , <<
  - **Bitwise Logical Operators**
    Keywords :
    ∼ , | , & , ∧

- **User defined types**
  - **Structures** : A structure is a user defined data type made up of variables of other data types.

Syntax :
struct point {
    <define variables>;
};

In the above given example of structures, struct is the keyword used to define structures. "point" is the name of the structure which we want to assign and we declare variables in structures using the data types by enclosing them in curly braces.

- **Arrays** : Array is a data structure that store one or more elements consecutively in memory. Indexing of the array will start from 0. For defining the array we will define the array with constant size specifying the size initially.
  - Array Declaration : When we declare an array we first define its data type which can be all the integer, double, float, character data types which we had defined earlier in this document.
    Syntax :
    <data type> arr[<array size>];
  - Array Initialisation : It is done to define the elements of the initialised array.

- **Functions** (Recursion supported) : To separate parts of your program into distinct sub procedures.
  - Function Declaration : Function declaration is done by specifying the function name, return type of the function and the parameter list.
    Syntax :
    <return-type> <function-name> (parameter-list);
  - Function Definition : By defining the function, we determine that what the function will do.
    Syntax :
    <return-type> <function-name> (parameter-list) {
        <function body>;
    }
  - Function Calling : If we want to use a declared function we call it as follows.
    Syntax :
    <function-name> (parameter-list);

- Recursive Function : These are those function which calls them-self in their definition.

- **Variable** : A variable is a named location in memory. Variables are used to give a value a name so we can refer to it later.
  - Variable Declaration : A section of code enclosed in brackets and form a block. All variables must be declared before they are used. Variables must be declared at the top of a block before any statements. They may be initialized by a constant or an expression when declared.

    Syntax :
    <type> <variable-name>;
    variable-name = initial-value;

- **Expressions** : An expression is a statement that has a value – for instance, a number, a string, the sum of two numbers, etc. Expressions combine variables and constants to produce new values. For example, 4+2, x-1, and "Hello, world!" are all expressions.

- **Input/Output statements**
  - **Input** :
    Keyword : **scanf** : scanf is a function that reads formatted data from stdin (i.e, the standard input stream, which is usually the keyboard, unless redirected) and then writes the results into the arguments given.

    Syntax :scanf("format string",argument list);
  - **Output** :
    Keyword : **printf** : printf() function is used for output. It prints the given statement to the console.

    Syntax : printf("format string",argument list);

- **Control Structures**
  - **Conditionals** :
    * **if** : An if statement consists of a Boolean expression followed by one or more statements.
    * **if-else** : An if statement can be followed by an optional else statement, which executes when the Boolean expression is false.
    * **nested if** : Can use one if or else if statement inside another if or else if statement(s).
    * **switch** : A switch statement allows a variable to be tested for equality against a list of values.
    * **Switch nested** : Can use one switch statement inside another switch statement(s).
    * **Conditional Operator** : The conditional operator is kind of similar to the if-else statement as it does follow the same algorithm as of if-else statement but the conditional operator takes less space and helps to write the if-else statements in the shortest way possible.
      Syntax :
      variable = Expression1 ? Expression2 : Expression3
  - **Loops** :
    A loop statement allows us to execute a statement or group of statements multiple times.
    * **for** : A for loop is a repetition control structure that allows us to efficiently write a loop that needs to execute a specific number of times .
      Syntax :
      for (init; condition; increment) {
      statement(s);
      }

* **While** : A while loop in C programming repeatedly executes a target statement as long as a given condition is true.
  Syntax :
  while(condition) {
  statement(s);
  }

* **Do-While** : Unlike for and while loops, which test the loop condition at the top of the loop, the do...while loop in C programming checks its condition at the bottom of the loop.
  Syntax :
  do {
  statement(s);
  }
  while(condition);

- **Pointers** : The pointer in is a variable which stores the address of another variable. This variable can be of type int, char, array, function, or any other pointer. The size of the pointer depends on the architecture.

  Pointer Declaration : The pointer in can be declared using * (asterisk symbol). It is also known as indirection pointer used to de-reference a pointer.
  Syntax :
  datatype * pointer-name;

Advanced Features :

- **Multi-level Pointers** : A pointer is pointer to another pointer which can be pointer to others pointers and so on is know as multilevel pointers. In our implementation, it will support upto level 2 means a pointer pointing to another pointer.

- **Function Pointers** : Store the address of the function.
  Syntax for declaration :
  <return datatype> (*<pointer name>) (parameter-list);
  - Address of a function **:** We can also get the address of the function by dereferencing it.
    Syntax :
    &<function name>
- **Simple library functions** such as,
  - **String Functions** : String functions are used on strings to perform some operations directly which makes out tasks easy.
    * **length()**: This function return the length of the string.
      Syntax to use :
      <name of string>**.**length()
    * **strcmp(str1,str2)**: Returns -ve value if str1 is less than str2 and 0 if str1 is equal to str2 and >0 (+ve value) if str1 is greater than str2.
      Syntax to use :
      strcmp(<string name>,<string name>)
    * **strcat(str1,str2)** : Append the string str2 to the str1.
      This function returns dest, the pointer to the destination string
      Syntax to use :
      strcat(<string name>,<string name>)
    * **strcpy(str1,str2)** : It will copy the content of string "str2" into the "str1"
      It returns the pointer to the destination.

Syntax to use :
strcpy(<destination string>,<source string>)

- **Math Functions** : Math functions are used to do some frequently used operations.
  * **pow()** : This function takes base and exponent as arguments and returns base raised to the power of exponent.
    Syntax to use :
    pow(<base>,<exponent>)

  * **sqrt()** : This function takes numeric value as arguments and returns the square root of value
    Syntax to use :
    sqrt(<number>)

  * **max()** : It takes two parameters and returns the biggest of these values.
    Syntax to use :
    max(<value1>,<value2>)

  * **min()**: It takes two parameters and returns the minimum of these values.
    Syntax to use :
    min(<value1>,<value2>)

  * **ceil()**: returns the smallest integer that is greater than or equal to the passed argument
    Syntax to use :
    ceil(<number>)

  * **floor()**: returns the smallest integer that is smaller than or equal to the passed argument
    Syntax to use :
    floor(<number>)

- **Dynamic Memory Allocation/Management** : Dynamic memory allocation in C++ refers to performing memory allocation manually by programmer.
  Keywords : new, delete

- **new** : The new operator denotes a request for memory allocation. If sufficient memory is available, new operator initializes the memory and returns the address of the newly allocated and initialized memory to the pointer variable.
  Syntax to use :
  pointer-variable = new <data-type>

- **delete** : to deallocate dynamically allocated memory, programmers are provided delete operator
  Syntax to use :
  delete <pointer-variable>

- **Dead Code Eliminaton** : our compiler will do the optimization, It removes the dead code(which is not usefull or not effecting the execution).

References :

- https://www.gnu.org/software/gnu-c-manual/gnu-c-manual.html
- https://www.geeksforgeeks.org/c-data-types/
- https://www.cplusplus.com/files/tutorial.pdf
- https://www.geeksforgeeks.org/new-and-delete-operators-in-cpp-for-dynamic-memory/
- https://en.wikipedia.org/wiki/Dead_code_elimination