Computer languages and compilation process

Department of Computer Science
City University of New York, Brooklyn College

Computer languages and compilation process

September 2010

Outline

Outline

Machine Languages I

- The language computer's CPU can understand
- Programmers control a computer by defining instructions for its CPU ; can be done either by writing programs directly in machine language or writing programs in a high-level language then translating them into machine language
- Machine language is a set of binary codes corresponding to the basic tasks that a CPU can perform

Machine language Example

- 1010000100 RR RR RR
- add contents of two registers, store result in another register

Machine Languages II

- CPU carries out the machine language instructions.
- CPU has 3 parts : ALU, Registers, CU
- ALU: Arithmetic Logic Unit does actual operations on data such as addition, subtraction
- Registers : Memory Locations that are built into the CPU
- CU : (Control Unit) Brain within the brain , fetches data and instructions from main memory, control the flow of data from registers to the ALU and back o registers

Outline

Assembly Languages

Example

- SET r1, 10 ; set r1 to immediate value 10
- STORE X, r1 ; store r1 into variable x

- Machine languages consist entirely of numbers and are almost impossible for humans to read and write.
- Assembly languages have the same structure and set of commands as machine languages.
- But they enable a programmer to use names instead of numbers.

Assembly Languages

Example

- SET r1, 10 ; set r1 to immediate value 10
- STORE X, r1 ; store r1 into variable x

- Machine languages consist entirely of numbers and are almost impossible for humans to read and write.
- Assembly languages have the same structure and set of commands as machine languages.
- But they enable a programmer to use names instead of numbers.

Assembly Languages

Example

- SET r1, 10 ; set r1 to immediate value 10
- STORE X, r1 ; store r1 into variable x

- Machine languages consist entirely of numbers and are almost impossible for humans to read and write.
- Assembly languages have the same structure and set of commands as machine languages.
- But they enable a programmer to use names instead of numbers.

Assembly Languages

Example

- SET r1, 10 ; set r1 to immediate value 10
- STORE X, r1 ; store r1 into variable x

- Machine languages consist entirely of numbers and are almost impossible for humans to read and write.
- Assembly languages have the same structure and set of commands as machine languages.
- But they enable a programmer to use names instead of numbers.

Assembly Languages

Example

- SET r1, 10 ; set r1 to immediate value 10
- STORE X, r1 ; store r1 into variable x

- Machine languages consist entirely of numbers and are almost impossible for humans to read and write.
- Assembly languages have the same structure and set of commands as machine languages.
- But they enable a programmer to use names instead of numbers.

Outline

High Level Programming Languages

Example

```
OPEN(6,FILE='PRN')
SUM1=.9*(1.-0.1**6)/0.9
SUM2=.9*(1.-0.1**7)/0.9
DIF = SUM2 - SUM1
```

- Interaction with the computer had to become simpler.
- Computer tasks are specified via mathematical formulas or words

High Level Programming Languages

Example

```
OPEN(6,FILE='PRN')
SUM1=.9*(1.-0.1**6)/0.9
SUM2=.9*(1.-0.1**7)/0.9
DIF = SUM2 - SUM1
```

- Interaction with the computer had to become simpler.
- Computer tasks are specified via mathematical formulas or words

Programming Paradigms

- A variable is a named container for data
- Data type is the description of the kinds of data stored, passed and used such as Integer, Decimal, Character, String
- Control statements change the computer's control from automatically reading the next line of code to reading a different one
- Assignment statements sets or re-sets the value stored in the storage location(s) denoted by a variable name

- Int num1, num2 :: Two variables, their data type is Integer
- num1 = 5 :: Assignment statement
- if (num1 > 0) num2 = 5 else num2 = 10 :: Control statement

Programming Paradigms

- A variable is a named container for data
- Data type is the description of the kinds of data stored, passed and used such as Integer, Decimal, Character, String
- Control statements change the computer's control from automatically reading the next line of code to reading a different one
- Assignment statements sets or re-sets the value stored in the storage location(s) denoted by a variable name

- Int num1, num2 :: Two variables, their data type is Integer
- num1 = 5 :: Assignment statement
- if (num1 > 0) num2 = 5 else num2 = 10 :: Control statement

Programming Paradigms

- A variable is a named container for data
- Data type is the description of the kinds of data stored, passed and used such as Integer, Decimal, Character, String
- Control statements change the computer's control from automatically reading the next line of code to reading a different one
- Assignment statements sets or re-sets the value stored in the storage location(s) denoted by a variable name

- Int num1, num2 :: Two variables, their data type is Integer
- num1 = 5 :: Assignment statement
- if (num1 > 0) num2 = 5 else num2 = 10 :: Control statement

Programming Paradigms

- A variable is a named container for data
- Data type is the description of the kinds of data stored, passed and used such as Integer, Decimal, Character, String
- Control statements change the computer's control from automatically reading the next line of code to reading a different one
- Assignment statements sets or re-sets the value stored in the storage location(s) denoted by a variable name

- Int num1, num2 :: Two variables, their data type is Integer
- num1 = 5 :: Assignment statement
- if (num1 > 0) num2 = 5 else num2 = 10 :: Control statement

Programming Paradigms

- A variable is a named container for data
- Data type is the description of the kinds of data stored, passed and used such as Integer, Decimal, Character, String
- Control statements change the computer's control from automatically reading the next line of code to reading a different one
- Assignment statements sets or re-sets the value stored in the storage location(s) denoted by a variable name

- Int num1, num2 :: Two variables, their data type is Integer
- num1 = 5 :: Assignment statement
- if (num1 > 0) num2 = 5 else num2 = 10 :: Control statement

Programming Paradigms

- A variable is a named container for data
- Data type is the description of the kinds of data stored, passed and used such as Integer, Decimal, Character, String
- Control statements change the computer's control from automatically reading the next line of code to reading a different one
- Assignment statements sets or re-sets the value stored in the storage location(s) denoted by a variable name

- Int num1, num2 :: Two variables, their data type is Integer
- num1 = 5 :: Assignment statement
- if (num1 > 0) num2 = 5 else num2 = 10 :: Control statement

Pseudocode

- used to outline the general steps in an algorithm without having to write actual code (usually done for the reader's or programmer's benefit)

Data types	Integer sum, decimal average, integer grade1, integer grade2
Loop	Do the following 5 times
Statement	Read grade1 and grade2 from user
Assignment statement	Sum=sum+grade1
Assignment statement	Sum=sum+grade2
Assignment statement	Average=sum/2
Statement	Print average
Statement	Print "done 5 times"

Compilation Process

- compilation is translating a high level language program into machine language
- the resulting machine language program can then be executed directly and repeatedly on the computer
- although they are all converted to machine language, different programming languages have different syntax rules

Some C code

```
int main(void){
int sum;
double average;
int grade1, grade2, loopcount;
loopcount=5;
while(loopcount>0){
scanf(''%d %d'', &grade1,&grade2);
sum = sum + grade1;
sum = sum + grade2;
average = sum/2;
printf(''%d'', average);
loopcount = loopcount - 1; }
return;
}
```

Some Java code

```
public class , AverageOfTwoNumbers {
public static void main(String[] args){
double num1=0.0;
double num2=0.0;
double sum=0.0;
double average=0.0;
BufferedReader dataIn=new BufferedReader( new InputS
try{
System.out.print("Enter Num 1: ");
numl=Double.parseDouble(dataIn.readLine());
System.out.print("Enter Num 2: ");
Num2=Double.parseDouble(dataIn.readLine());
average=n1+n2/2;
System.out.print(average);
}catch(IOException e){
```

ASCII codes

ASCII American Standard Code for Information Interchange

- Computers can manage internally only 0s (zeros) and 1s (ones).
- There should be a way to represent letters and other non-numeric characters with 0s and 1s.
- Computers use ASCII tables, which are tables or lists that contain all the letters in the roman alphabet plus some additional characters.
- Each character is always represented by the same order number.
- the ASCII code for the capital letter A is 65, which is represented using 0s and 1s in binary: 1000001.
- The standard ASCII table defines 128 character codes (from 0 to 127), of which, the first 32 are control codes

ASCII codes

ASCII American Standard Code for Information Interchange

- Computers can manage internally only 0s (zeros) and 1s (ones).
- There should be a way to represent letters and other non-numeric characters with 0s and 1s.
- Computers use ASCII tables, which are tables or lists that contain all the letters in the roman alphabet plus some additional characters.
- Each character is always represented by the same order number.
- the ASCII code for the capital letter A is 65, which is represented using 0s and 1s in binary: 1000001.
- The standard ASCII table defines 128 character codes (from 0 to 127), of which, the first 32 are control codes

ASCII codes

ASCII American Standard Code for Information Interchange

- Computers can manage internally only 0s (zeros) and 1s (ones).
- There should be a way to represent letters and other non-numeric characters with 0s and 1s.
- Computers use ASCII tables, which are tables or lists that contain all the letters in the roman alphabet plus some additional characters.
- Each character is always represented by the same order number.
- the ASCII code for the capital letter A is 65, which is represented using 0s and 1s in binary: 1000001.
- The standard ASCII table defines 128 character codes (from 0 to 127), of which, the first 32 are control codes

ASCII codes

ASCII American Standard Code for Information Interchange

- Computers can manage internally only 0s (zeros) and 1s (ones).
- There should be a way to represent letters and other non-numeric characters with 0s and 1s.
- Computers use ASCII tables, which are tables or lists that contain all the letters in the roman alphabet plus some additional characters.
- Each character is always represented by the same order number.
- the ASCII code for the capital letter A is 65, which is represented using 0s and 1s in binary: 1000001.
- The standard ASCII table defines 128 character codes (from 0 to 127), of which, the first 32 are control codes

ASCII codes

ASCII American Standard Code for Information Interchange

- Computers can manage internally only 0s (zeros) and 1s (ones).
- There should be a way to represent letters and other non-numeric characters with 0s and 1s.
- Computers use ASCII tables, which are tables or lists that contain all the letters in the roman alphabet plus some additional characters.
- Each character is always represented by the same order number.
- the ASCII code for the capital letter A is 65, which is represented using 0s and 1s in binary: 1000001.
- The standard ASCII table defines 128 character codes (from 0 to 127), of which, the first 32 are control codes

ASCII codes

ASCII American Standard Code for Information Interchange

- Computers can manage internally only 0s (zeros) and 1s (ones).
- There should be a way to represent letters and other non-numeric characters with 0s and 1s.
- Computers use ASCII tables, which are tables or lists that contain all the letters in the roman alphabet plus some additional characters.
- Each character is always represented by the same order number.
- the ASCII code for the capital letter A is 65, which is represented using 0s and 1s in binary: 1000001.
- The standard ASCII table defines 128 character codes (from 0 to 127), of which, the first 32 are control codes

ASCII codes

ASCII American Standard Code for Information Interchange

- Computers can manage internally only 0s (zeros) and 1s (ones).
- There should be a way to represent letters and other non-numeric characters with 0s and 1s.
- Computers use ASCII tables, which are tables or lists that contain all the letters in the roman alphabet plus some additional characters.
- Each character is always represented by the same order number.
- the ASCII code for the capital letter A is 65, which is represented using 0s and 1s in binary: 1000001.
- The standard ASCII table defines 128 character codes (from 0 to 127), of which, the first 32 are control codes

reminders

- There will be Exam 1 on October 11.
- It will be similar to the Sample exam you have
- Reading only lecture slides is not enough
- You should also read the chapters listed on the syllabus on the web site
- My office hour is Monday 5:30-6:30, 128NE(New Ingersoll Building)
- If you want to come to my office hour, please email me and make appointment with me two class days before

Thank you for your attention