Basics continued!

Variables and arithmetic

Professor Daniel Leeds
dleeds@fordham.edu
JMH 328A

CISC 1600/1610
Computer Science I

Variables store information

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
<td>single character (‘a’, ‘Q’)</td>
</tr>
<tr>
<td>int</td>
<td>integers (-4, 82)</td>
</tr>
<tr>
<td>bool</td>
<td>logic (true or false)</td>
</tr>
<tr>
<td>float</td>
<td>real numbers (1.3, -0.45)</td>
</tr>
<tr>
<td>string</td>
<td>text (&quot;Hello&quot;, &quot;reload&quot;)</td>
</tr>
</tbody>
</table>

Variable declaration

```c++
extern "C" int add(int a, int b)
{
    return a + b;
}
```

Variable declaration and initialization

- All variables must be declared before they are used
- Variables are initialized with the first assignment statement
- Declaration and initialization can be performed in one line

```c++
int cost;  // declare variable
```

User input: “Hello _____!”

```c++
#include <iostream>
using namespace std;

int main()
{
    // Begin main function
    string name;  // create variable called name
    cout << "What is your name?";
    cin >> name;  // get name from user
    cout << "Hello " << name << endl;  // output "Hello <name>"
    return 0;  // end program
}  // End main function
```

```
> ./myProgram
What is your name? Alice
Hello Alice!
```

Variable assignment

- Typically, variables are assigned values with the = operator
- The variable to be changed is always to the left of the = operator
- The value assigned from the right of the = operator

```c++
int ageErica = ageJen;
```

```
> ./myProgram
What is your name? Alice
Hello Alice!
```
Input command

```c++
#include <iostream>

using namespace std;

int main()
{
    // Begin main function
    string name;            // create variable called name
    cout << "What is your name?"; // output "What is your name?"
    cin >> name;             // get name from user
    cout << "Hello ";    // output "Hello 
    cout << name << "!"; // output "<name>!
    return 0;               // end program
}
```

Variable names

- A variable name is any valid identifier that is not a keyword
  - Starts with a letter, contains letters, digits, and underscores
  - Cannot begin with a digit
  - Case sensitive: `username` ≠ `UserName`

Variable names, part 2

Choose meaningful names
- Avoid acronyms
- Avoid lengthy names

Good:
- age, size, address, count, sumData
- $x, y, i$ - single letters as counting variables

Bad:
- rbi, lda, x25
- neuron_response_magnitude

Keywords

Also known as: “Reserved names”
- Examples
  - `cout`, `return`, `string`, `int`
- Must be used as they are defined in the programming language
- Cannot be used as variable names

Arithmetic in C++

Operators
- Addition: $5 + 2$ evaluates to $7$
- Subtraction: $5 - 2$ evaluates to $3$
- Multiplication: $5 * 2$ evaluates to $10$
- Division: $4 / 2$ evaluates to $2$
- Modulo: $5 \% 2$ evaluates to $1$ (only integers)

What does this program do?

```c++
#include <iostream>

using namespace std;

int main()
{
    int dollars, coins;
    cout << "How many dollars do you have? ";
    cin >> dollars;
    coins = dollars * 4;
    cout << "I will give you " << coins << " coins."
    return 0;
}
```
Order of operations
• First: Parentheses
• Second: Multiplication, Division, Modulo
• Third: Add, Subtract

- Evaluate from Left to Right
- Evaluate inner-most parentheses before outer ones

```
int a = ( 4 * ( 5 + 2 ) - 4 ) / 4;
```

Assignment operators
• int a = 6;
• Standard assignment: a = 3;
• Other assignments:
  - a += 3;  // a = a + 3;
  - a -= 3;  // a = a - 3;
  - a *= 3;  // a = a * 3;
  - a /= 3;  // a = a / 3;
  - a %= 3;  // a = a % 3;

Increment and decrement
```
int c = 12;
```
• Increment by 1: c++ evaluates to c + 1
• Decrement by 1: c-- evaluates to c - 1

The binary representation
• int age = 65; assigns a binary code to memory: 01000001
• char grade = 'A'; assigns a binary code to memory: 01000001
• Every variable value is a number in binary, C++ interprets the binary number based on the variable type

Interpreting binary

**Base 10**
253 -> 253
2x100+5x10+3x1

**Base 2**
128 64 32 16 8 4 2 1
- - - - - - -

- 00001001=?
- 00110000=?
- 10010010=?

From numbers to symbols:
the ASCII table

<table>
<thead>
<tr>
<th>Numeric Code</th>
<th>Character</th>
<th>Numeric Code</th>
<th>Character</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>space</td>
<td>65</td>
<td>A</td>
</tr>
<tr>
<td>40</td>
<td>(</td>
<td>69</td>
<td>B</td>
</tr>
<tr>
<td>44</td>
<td>,</td>
<td>72</td>
<td>C</td>
</tr>
<tr>
<td>47</td>
<td>?</td>
<td>73</td>
<td>D</td>
</tr>
<tr>
<td>48</td>
<td>0</td>
<td>74</td>
<td>E</td>
</tr>
<tr>
<td>49</td>
<td>1</td>
<td>75</td>
<td>F</td>
</tr>
<tr>
<td>52</td>
<td>2</td>
<td>76</td>
<td>G</td>
</tr>
<tr>
<td>53</td>
<td>3</td>
<td>77</td>
<td>H</td>
</tr>
<tr>
<td>54</td>
<td>4</td>
<td>78</td>
<td>I</td>
</tr>
<tr>
<td>55</td>
<td>5</td>
<td>79</td>
<td>J</td>
</tr>
<tr>
<td>56</td>
<td>6</td>
<td>80</td>
<td>K</td>
</tr>
<tr>
<td>57</td>
<td>7</td>
<td>81</td>
<td>L</td>
</tr>
<tr>
<td>58</td>
<td>8</td>
<td>82</td>
<td>M</td>
</tr>
<tr>
<td>59</td>
<td>9</td>
<td>83</td>
<td>N</td>
</tr>
<tr>
<td>62</td>
<td>:</td>
<td>85</td>
<td>P</td>
</tr>
<tr>
<td>63</td>
<td>;</td>
<td>86</td>
<td>Q</td>
</tr>
<tr>
<td>64</td>
<td>&lt;</td>
<td>87</td>
<td>R</td>
</tr>
<tr>
<td>65</td>
<td>=</td>
<td>88</td>
<td>S</td>
</tr>
<tr>
<td>66</td>
<td>&gt;</td>
<td>89</td>
<td>T</td>
</tr>
<tr>
<td>67</td>
<td>?</td>
<td>90</td>
<td>U</td>
</tr>
<tr>
<td>97</td>
<td>`</td>
<td>96</td>
<td>V</td>
</tr>
<tr>
<td>99</td>
<td>^</td>
<td>97</td>
<td>W</td>
</tr>
<tr>
<td>100</td>
<td>_</td>
<td>98</td>
<td>X</td>
</tr>
<tr>
<td>101</td>
<td>`</td>
<td>99</td>
<td>Y</td>
</tr>
<tr>
<td>102</td>
<td>'</td>
<td>100</td>
<td>Z</td>
</tr>
<tr>
<td>103</td>
<td>`</td>
<td>101</td>
<td>[</td>
</tr>
<tr>
<td>104</td>
<td>`</td>
<td>102</td>
<td>]</td>
</tr>
<tr>
<td>105</td>
<td>`</td>
<td>103</td>
<td>{</td>
</tr>
<tr>
<td>106</td>
<td>`</td>
<td>104</td>
<td>}</td>
</tr>
<tr>
<td>107</td>
<td>`</td>
<td>105</td>
<td>~</td>
</tr>
<tr>
<td>123</td>
<td>a</td>
<td>124</td>
<td>b</td>
</tr>
<tr>
<td>125</td>
<td>b</td>
<td>126</td>
<td>c</td>
</tr>
<tr>
<td>127</td>
<td>c</td>
<td>128</td>
<td>d</td>
</tr>
<tr>
<td>129</td>
<td>d</td>
<td>130</td>
<td>e</td>
</tr>
<tr>
<td>131</td>
<td>e</td>
<td>132</td>
<td>f</td>
</tr>
<tr>
<td>133</td>
<td>f</td>
<td>134</td>
<td>g</td>
</tr>
<tr>
<td>135</td>
<td>g</td>
<td>136</td>
<td>h</td>
</tr>
<tr>
<td>137</td>
<td>h</td>
<td>138</td>
<td>i</td>
</tr>
<tr>
<td>139</td>
<td>i</td>
<td>140</td>
<td>j</td>
</tr>
<tr>
<td>141</td>
<td>j</td>
<td>142</td>
<td>k</td>
</tr>
<tr>
<td>143</td>
<td>k</td>
<td>144</td>
<td>l</td>
</tr>
<tr>
<td>145</td>
<td>l</td>
<td>146</td>
<td>m</td>
</tr>
<tr>
<td>147</td>
<td>m</td>
<td>148</td>
<td>n</td>
</tr>
<tr>
<td>149</td>
<td>n</td>
<td>150</td>
<td>o</td>
</tr>
<tr>
<td>151</td>
<td>o</td>
<td>152</td>
<td>p</td>
</tr>
<tr>
<td>153</td>
<td>p</td>
<td>154</td>
<td>q</td>
</tr>
<tr>
<td>155</td>
<td>q</td>
<td>156</td>
<td>r</td>
</tr>
<tr>
<td>157</td>
<td>r</td>
<td>158</td>
<td>s</td>
</tr>
<tr>
<td>159</td>
<td>s</td>
<td>160</td>
<td>t</td>
</tr>
<tr>
<td>161</td>
<td>t</td>
<td>162</td>
<td>u</td>
</tr>
<tr>
<td>163</td>
<td>u</td>
<td>164</td>
<td>v</td>
</tr>
<tr>
<td>165</td>
<td>v</td>
<td>166</td>
<td>w</td>
</tr>
<tr>
<td>167</td>
<td>w</td>
<td>168</td>
<td>x</td>
</tr>
<tr>
<td>169</td>
<td>x</td>
<td>170</td>
<td>y</td>
</tr>
<tr>
<td>171</td>
<td>y</td>
<td>172</td>
<td>z</td>
</tr>
<tr>
<td>173</td>
<td>z</td>
<td>174</td>
<td>[</td>
</tr>
<tr>
<td>175</td>
<td>[</td>
<td>176</td>
<td>]</td>
</tr>
<tr>
<td>177</td>
<td>]</td>
<td>178</td>
<td>{</td>
</tr>
<tr>
<td>179</td>
<td>{</td>
<td>180</td>
<td>}</td>
</tr>
<tr>
<td>181</td>
<td>}</td>
<td>182</td>
<td>~</td>
</tr>
</tbody>
</table>

1/25/2016
Variable types, revisited

<table>
<thead>
<tr>
<th>Datatype</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
<td>single character ('a', 'Q')</td>
<td>1 byte</td>
</tr>
<tr>
<td>int</td>
<td>integers (-4, 82)</td>
<td>4 bytes</td>
</tr>
<tr>
<td>bool</td>
<td>logic (true or false)</td>
<td>1 byte</td>
</tr>
<tr>
<td>float</td>
<td>real numbers (1.3, -0.45)</td>
<td>4 bytes</td>
</tr>
<tr>
<td>string</td>
<td>text (&quot;Hello&quot;, &quot;reload&quot;)</td>
<td>? bytes</td>
</tr>
</tbody>
</table>

- Each variable is represented by a certain number of 0s and 1s
- Each 0-or-1 is a bit
- 8 bits in a row is a byte

Assigning between types

```plaintext
int x = 5;
float y = -2.5;
float z = x * y;
int g = y - x;
```

Variables – locations in memory

- Each variable indicates a location in memory
- Each location holds a value
- Value can change as program progresses
- Variable value exists before initialization

<table>
<thead>
<tr>
<th>Address</th>
<th>Grade</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>04201320</td>
<td>A</td>
<td>140</td>
</tr>
<tr>
<td>04201328</td>
<td></td>
<td></td>
</tr>
<tr>
<td>04201336</td>
<td></td>
<td></td>
</tr>
<tr>
<td>04201344</td>
<td></td>
<td></td>
</tr>
<tr>
<td>04201352</td>
<td></td>
<td></td>
</tr>
<tr>
<td>04201360</td>
<td></td>
<td></td>
</tr>
<tr>
<td>04201368</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>