Perl for Biologists

Session 1
March 12, 2014

Introduction

Jaroslaw Pillardy
Organization

- Perl for Biologists consists of 15 sessions, one every week, until June 18th

- Sessions will be taught by different Bioinformatics Facility staff members, the speakers are listed on the workshop web pages

- Slides will be posted online before each session.

- Please feel free to contact us with any questions:
  - Workshop coordinator: Jaroslaw Pillardy jp86@cornell.edu, Rhodes 623
  - Each session’s speaker name is listed on session web page
  - You can find us in the CBSU directory http://cbsu.tc.cornell.edu/staff.aspx

- You can carry out practical exercises on your own machine/laptop/desktop or use our BioHPC Lab workstations allocated for you. Machine allocations are posted online on workshop pages http://cbsu.tc.cornell.edu/ww/1/Default.aspx?wid=46

- No programming experience necessary.
Organization

• BioHPC Lab machines are reserved for you and available all the time between now (March 12th) and June 29th (end of day June 28th)

• Please DO NOT use them for extensive calculations. They are set aside for you to learn Perl, not to carry out bioinformatics lab work 😊. You can always reserve more.

• You can see your reservations after logging into BioHPC Lab website http://cbsu.tc.cornell.edu/

• Helpful links:
  o Lab Users guide http://cbsu.tc.cornell.edu/lab/use.aspx
  o My reservations http://cbsu.tc.cornell.edu/lab/labresman.aspx
  o Reset password http://cbsu.tc.cornell.edu/lab/labpassreset.aspx

• Useful books:
  o “Learning Perl”, Randal Schwartz, Brain D Foy, Tom Phoenix
  o “Beginning Perl for Bioinformatics”, James Tisdall
Organization

• The workshop has practical examples and exercises.

• You can follow examples during the lecture, or you can carry them out afterwards.

• If you have any problems with them contact us or come to office hours

• **The only way to learn programming is to try!** Please do after lecture exercises – they are always discussed at the beginning of the next session.

• You can practice Perl programming on any computer, including your Windows or Mac laptop.

• We will focus on our Linux machines since it is most likely environment on which you will run your future Perl programs.

• Therefore next few slides are “Linux primer”.
Interacting with Linux: Terminal window

- User communicates with Linux machine via **commands** typed in the **terminal window**

  - Commands are interpreted by a program referred to as **shell** – an interface between Linux and the user. We will be using the shell called **bash** (another popular shell is **tcsh**).
  
  - Typically, each command is typed in one line and “**entered**” by hitting the **Enter** key on the keyboard.

  - Commands deal with **files** and **processes**, e.g.,
    - request information (e.g., list user’s files)
    - launch a simple task (e.g., rename a file)
    - start an application (e.g., Firefox web browser, BWA aligner, IGV viewer, …)
    - stop an application
Logging in to a Linux machine

- Details of the login procedure will depend on the infrastructure the machine is a part of

- On any Linux machine, you need
  - network name of the machine (e.g., cbsumm15.tc.cornell.edu)
  - an account, i.e., user ID and password
  - on your laptop: remote access software (typically: ssh client, VNC client)

- Linux is a multi-access, multi-tasking system: multiple users may be logged in and run multiple tasks on one machine at the same time, sharing resources (CPUs, memory, disk space)
  - This is what is happening during this workshop
Logging in to a Linux machine

- **BioHPC Lab-specific issues:**
  - User ID (normally: your Cornell NetID) and password have been sent to you by e-mail

![Image](image.png)

- Normally, you need a machine reservation
  - A reservation has been made for you on one of the machines for the workshop (will last through June 28)
- Forgot your password? Re-set it at [http://cbsu.tc.cornell.edu/lab/labpassreset.aspx](http://cbsu.tc.cornell.edu/lab/labpassreset.aspx)
How to access BioHPC Lab machines in the future (after workshop)

BioHPC Lab User’s Guide
http://cbsu.tc.cornell.edu/lab/userguide.aspx

Slides from workshop “Introduction to BioHPC Lab”

Slides from workshop “Linux for Biologists”
http://cbsu.tc.cornell.edu/lab/doc/Linux_workshop_Part2.pdf
Logging in to a Linux machine

- From a Windows PC via *ssh client* (ssh=Secure Shell)

  - Install remote access software (*PuTTY, MobaXterm*). For details, consult [http://cbsu.tc.cornell.edu/lab/doc/Remote_access.pdf](http://cbsu.tc.cornell.edu/lab/doc/Remote_access.pdf)

  - Use **PuTTY** to open a terminal window on the reserved workstation using **ssh** protocol, configure **X11 forwarding** (if you intend to run graphical software)
    - NOTE: while you are typing your password, the terminal will appear frozen – this is on purpose!

  - You may open several terminal windows, if needed (in PuTTY – can use “Duplicate Session” function).
Logging in to a Linux machine

From other Linux machine or Mac via native ssh client

- Launch the Mac’s terminal window and type

  `ssh -Y bukowski@cbsuwrkstX.tc.cornell.edu`

  (replace the “cbsuwrkstX” with the workstation that you just reserved, and “bukowski” with your own user ID). Enter the lab password when prompted.

  - NOTE: while you are typing your password, the terminal will appear frozen – this is on purpose!

- You may open several terminal windows, if needed, and log in to the workstation from each of them.
Logging **out** of a Linux machine

- While in terminal window, type `exit` or `Ctrl-d` - this will close the current terminal window.
Logging in to BioHPC Lab machines from outside of Cornell

Two ways to connect from outside:

- Install and run the CIT-recommended software to join the Cornell network, then proceed as usual.

- Log in to cbsulogin.tc.cornell.edu:

  \texttt{ssh jarekp@cbsulogin.tc.cornell.edu} \quad (\text{using PuTTy or other ssh client program})

  Once logged in to cbsulogin, ssh further to your reserved machine:

  \texttt{ssh jarekp@cbsuwrkst3.tc.cornell.edu}

Backup login proxy server is cbsuss02.tc.cornell.edu
Programming languages

• **Strongly typed** vs. **Loosely typed (context based)**
  - All variables declared
  - C, C++, Java, C#
  - Variables interpreted dynamically
  - Perl, Python, Visual Basic

• **Scripted (interpreted)** vs. **Compiled**
  - Executed “on the fly”, by line
  - Perl, Visual Basic, Shell
  - Python, Java, C#
  - Binary version of code executed
  - C, C++, Fortran

• **Flat** vs. **Object oriented**
  - No complex objects
  - C, Pascal
  - Objects with properties and functions
  - Perl, Java, C#, C++
Programming languages

Perl is a loosely typed, interpreted, object-oriented programming language.

Loosely typed:
Easier to write, more flexible, no need for extra code to “cast” variables. VERY EASY to make errors. Perl variables are typed dynamically based on context.

Interpreted:
More portable – will execute anywhere where interpreter is present IF program does not require specific libraries and IF it doesn’t use system specific commands. MUCH slower, automatic code optimization impossible.

Object-oriented:
Program can be compartmentalized with reusable code. Very powerful way to solve problems. Slower.
Why Perl?

- Easy to learn, fast to write (rapid prototyping), informal
- High-level – compact code, lots of useful functions
- Huge public library of code available that can be directly used
- Runs anywhere (with some caution)
- Flexible: useful for scripting, websites as well as large programs
- Perl is not fast, but excellent to “stich” together other programs – very good for pipelines, task automation, interacting with OS.
- Perl can be easily used to perform various “in-between” functions like process control, file/data control and conversion, string operations, database operations and many more
Programming cycle

EDIT / DESIGN → VERIFY / COMPILE

RUN / TEST
Perl programs are scripts – text files interpreted line by line.

Need to use TEXT editor to create and edit them.

TEXT file is a file that uses only letters, numbers and common symbols plus “new line” or “tab” special characters. NO formatting or other binary code (MS Word vs. text example).

Plain ASCII characters: byte codes between 32 and 126.

Modern text files can use special characters (e.g. ó or ö) and symbols (e.g. β or §) with Unicode – and Perl can work with them too. But they MUST be used with a TEXT editor (and better yet – not used at all 😊).

Example: Notepad and Word
### ASCII Table

<table>
<thead>
<tr>
<th>Dec</th>
<th>Hx</th>
<th>Oct</th>
<th>Char</th>
<th>Dec</th>
<th>Hx</th>
<th>Oct</th>
<th>Html</th>
<th>Chr</th>
<th>Dec</th>
<th>Hx</th>
<th>Oct</th>
<th>Html</th>
<th>Chr</th>
<th>Dec</th>
<th>Hx</th>
<th>Oct</th>
<th>Html</th>
<th>Chr</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>000</td>
<td>NUL (null)</td>
<td>32</td>
<td>20</td>
<td>040</td>
<td>ä#32; Space</td>
<td>64</td>
<td>40</td>
<td>100</td>
<td>ä#64; ©</td>
<td>96</td>
<td>60</td>
<td>140</td>
<td>ä#96; `</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>001</td>
<td>SOH (start of heading)</td>
<td>33</td>
<td>21</td>
<td>041</td>
<td>ä#33; !</td>
<td>65</td>
<td>41</td>
<td>101</td>
<td>ä#65; A</td>
<td>97</td>
<td>61</td>
<td>141</td>
<td>ä#97; a</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>002</td>
<td>STX (start of text)</td>
<td>34</td>
<td>22</td>
<td>042</td>
<td>ä#34; &quot;</td>
<td>66</td>
<td>42</td>
<td>102</td>
<td>ä#66; B</td>
<td>98</td>
<td>62</td>
<td>142</td>
<td>ä#98; b</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>003</td>
<td>ETX (end of text)</td>
<td>35</td>
<td>23</td>
<td>043</td>
<td>ä#35; #</td>
<td>67</td>
<td>43</td>
<td>103</td>
<td>ä#67; C</td>
<td>99</td>
<td>63</td>
<td>143</td>
<td>ä#99; c</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>004</td>
<td>EOT (end of transmission)</td>
<td>36</td>
<td>24</td>
<td>044</td>
<td>ä#36; $</td>
<td>68</td>
<td>44</td>
<td>104</td>
<td>ä#68; D</td>
<td>100</td>
<td>64</td>
<td>144</td>
<td>ä#100; d</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>005</td>
<td>ENQ (enquiry)</td>
<td>37</td>
<td>25</td>
<td>045</td>
<td>ä#37; %</td>
<td>69</td>
<td>45</td>
<td>105</td>
<td>ä#69; E</td>
<td>101</td>
<td>65</td>
<td>145</td>
<td>ä#101; e</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>006</td>
<td>ACK (acknowledge)</td>
<td>38</td>
<td>26</td>
<td>046</td>
<td>ä#38; &amp;</td>
<td>70</td>
<td>46</td>
<td>106</td>
<td>ä#70; F</td>
<td>102</td>
<td>66</td>
<td>146</td>
<td>ä#102; f</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>007</td>
<td>BEL (bell)</td>
<td>39</td>
<td>27</td>
<td>047</td>
<td>ä#39; '</td>
<td>71</td>
<td>47</td>
<td>107</td>
<td>ä#71; G</td>
<td>103</td>
<td>67</td>
<td>147</td>
<td>ä#103; g</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>010</td>
<td>BS (backspace)</td>
<td>40</td>
<td>28</td>
<td>040</td>
<td>ä#40; (</td>
<td>72</td>
<td>48</td>
<td>110</td>
<td>ä#72; H</td>
<td>104</td>
<td>68</td>
<td>150</td>
<td>ä#104; h</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>011</td>
<td>TAB (horizontal tab)</td>
<td>41</td>
<td>29</td>
<td>051</td>
<td>ä#41; )</td>
<td>73</td>
<td>49</td>
<td>111</td>
<td>ä#73; I</td>
<td>105</td>
<td>69</td>
<td>151</td>
<td>ä#105; i</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>A</td>
<td>012</td>
<td>LF (NL line feed, new line)</td>
<td>42</td>
<td>3A</td>
<td>052</td>
<td>ä#42; *</td>
<td>74</td>
<td>4A</td>
<td>112</td>
<td>ä#74; J</td>
<td>106</td>
<td>6A</td>
<td>152</td>
<td>ä#106; j</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>013</td>
<td>VT (vertical tab)</td>
<td>43</td>
<td>3B</td>
<td>053</td>
<td>ä#43; +</td>
<td>75</td>
<td>4B</td>
<td>113</td>
<td>ä#75; K</td>
<td>107</td>
<td>6B</td>
<td>153</td>
<td>ä#107; k</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>C</td>
<td>014</td>
<td>FF (NP form feed, new page)</td>
<td>44</td>
<td>3C</td>
<td>054</td>
<td>ä#44; ,</td>
<td>76</td>
<td>4C</td>
<td>114</td>
<td>ä#76; L</td>
<td>108</td>
<td>6C</td>
<td>154</td>
<td>ä#108; l</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>D</td>
<td>015</td>
<td>CR (carriage return)</td>
<td>45</td>
<td>3D</td>
<td>055</td>
<td>ä#45; -</td>
<td>77</td>
<td>4D</td>
<td>115</td>
<td>ä#77; M</td>
<td>109</td>
<td>6D</td>
<td>155</td>
<td>ä#109; m</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td>016</td>
<td>SO (shift out)</td>
<td>46</td>
<td>3E</td>
<td>056</td>
<td>ä#46; .</td>
<td>78</td>
<td>4E</td>
<td>116</td>
<td>ä#78; N</td>
<td>110</td>
<td>6E</td>
<td>156</td>
<td>ä#110; n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td>017</td>
<td>SI (shift in)</td>
<td>47</td>
<td>3F</td>
<td>057</td>
<td>ä#47; /</td>
<td>79</td>
<td>4F</td>
<td>117</td>
<td>ä#79; O</td>
<td>111</td>
<td>6F</td>
<td>157</td>
<td>ä#111; o</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>10</td>
<td>020</td>
<td>DLE (data link escape)</td>
<td>48</td>
<td>30</td>
<td>060</td>
<td>ä#48; 0</td>
<td>80</td>
<td>50</td>
<td>120</td>
<td>ä#80; P</td>
<td>112</td>
<td>70</td>
<td>160</td>
<td>ä#112; p</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>021</td>
<td>DC1 (device control 1)</td>
<td>49</td>
<td>31</td>
<td>061</td>
<td>ä#49; 1</td>
<td>81</td>
<td>51</td>
<td>121</td>
<td>ä#81; Q</td>
<td>113</td>
<td>71</td>
<td>161</td>
<td>ä#113; q</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>022</td>
<td>DC2 (device control 2)</td>
<td>50</td>
<td>32</td>
<td>062</td>
<td>ä#50; 2</td>
<td>82</td>
<td>52</td>
<td>122</td>
<td>ä#82; R</td>
<td>114</td>
<td>72</td>
<td>162</td>
<td>ä#114; r</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>13</td>
<td>023</td>
<td>DC3 (device control 3)</td>
<td>51</td>
<td>33</td>
<td>063</td>
<td>ä#51; 3</td>
<td>83</td>
<td>53</td>
<td>123</td>
<td>ä#83; S</td>
<td>115</td>
<td>73</td>
<td>163</td>
<td>ä#115; s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>14</td>
<td>024</td>
<td>DC4 (device control 4)</td>
<td>52</td>
<td>34</td>
<td>064</td>
<td>ä#52; 4</td>
<td>84</td>
<td>54</td>
<td>124</td>
<td>ä#84; T</td>
<td>116</td>
<td>74</td>
<td>164</td>
<td>ä#116; t</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>15</td>
<td>025</td>
<td>NAK (negative acknowledge)</td>
<td>53</td>
<td>35</td>
<td>065</td>
<td>ä#53; 5</td>
<td>85</td>
<td>55</td>
<td>125</td>
<td>ä#85; U</td>
<td>117</td>
<td>75</td>
<td>165</td>
<td>ä#117; u</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>16</td>
<td>026</td>
<td>SYN (synchronous idle)</td>
<td>54</td>
<td>36</td>
<td>066</td>
<td>ä#54; 6</td>
<td>86</td>
<td>56</td>
<td>126</td>
<td>ä#86; V</td>
<td>118</td>
<td>76</td>
<td>166</td>
<td>ä#118; v</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>17</td>
<td>027</td>
<td>ETB (end of trans. block)</td>
<td>55</td>
<td>37</td>
<td>067</td>
<td>ä#55; 7</td>
<td>87</td>
<td>57</td>
<td>127</td>
<td>ä#87; W</td>
<td>119</td>
<td>77</td>
<td>167</td>
<td>ä#119; w</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>18</td>
<td>030</td>
<td>CAN (cancel)</td>
<td>56</td>
<td>38</td>
<td>070</td>
<td>ä#56; 8</td>
<td>88</td>
<td>58</td>
<td>130</td>
<td>ä#88; X</td>
<td>120</td>
<td>78</td>
<td>170</td>
<td>ä#120; x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>19</td>
<td>031</td>
<td>EM (end of medium)</td>
<td>57</td>
<td>39</td>
<td>071</td>
<td>ä#57; 9</td>
<td>89</td>
<td>59</td>
<td>131</td>
<td>ä#89; Y</td>
<td>121</td>
<td>79</td>
<td>171</td>
<td>ä#121; y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>1A</td>
<td>032</td>
<td>SUB (substitute)</td>
<td>58</td>
<td>3A</td>
<td>072</td>
<td>ä#58; :</td>
<td>90</td>
<td>5A</td>
<td>132</td>
<td>ä#90; Z</td>
<td>122</td>
<td>7A</td>
<td>172</td>
<td>ä#122; z</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>1B</td>
<td>033</td>
<td>ESC (escape)</td>
<td>59</td>
<td>3B</td>
<td>073</td>
<td>ä#59; ;</td>
<td>91</td>
<td>5B</td>
<td>133</td>
<td>ä#91; [</td>
<td>123</td>
<td>7B</td>
<td>173</td>
<td>ä#123; {</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>1C</td>
<td>034</td>
<td>FS (file separator)</td>
<td>60</td>
<td>3C</td>
<td>074</td>
<td>ä#60; &lt;</td>
<td>92</td>
<td>5C</td>
<td>134</td>
<td>ä#92; \</td>
<td>124</td>
<td>7C</td>
<td>174</td>
<td>ä#124;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1D</td>
<td>035</td>
<td>GS (group separator)</td>
<td>61</td>
<td>3D</td>
<td>075</td>
<td>ä#61; =</td>
<td>93</td>
<td>5D</td>
<td>135</td>
<td>ä#93; ]</td>
<td>125</td>
<td>7D</td>
<td>175</td>
<td>ä#125; }</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>1E</td>
<td>036</td>
<td>RS (record separator)</td>
<td>62</td>
<td>3E</td>
<td>076</td>
<td>ä#62; &gt;</td>
<td>94</td>
<td>5E</td>
<td>136</td>
<td>ä#94; ^</td>
<td>126</td>
<td>7E</td>
<td>176</td>
<td>ä#126; ~</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>1F</td>
<td>037</td>
<td>US (unit separator)</td>
<td>63</td>
<td>3F</td>
<td>077</td>
<td>ä#63; ?</td>
<td>95</td>
<td>5F</td>
<td>137</td>
<td>ä#95; _</td>
<td>127</td>
<td>7F</td>
<td>177</td>
<td>ä#127; DEL</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Source: www.LookupTables.com
### ASCII Table

| 128 | Ç   | 144 | É   | 160 | à   | 176 |  | 192 | ℐ | 208 |  |  | 224 | α | 240 | ≫ |
| 129 | ù   | 145 | æ   | 161 | í   | 177 |  | 193 | ℐ | 209 |  |  | 225 | β | 241 | ≫ |
| 130 | é   | 146 | Æ   | 162 | ó   | 178 |  | 194 | ℐ | 210 |  |  | 226 | Γ | 242 | ≫ |
| 131 | â   | 147 | ô   | 163 | ü   | 179 |  | 195 | ℐ | 211 |  |  | 227 | π | 243 | ≫ |
| 132 | ä   | 148 | ö   | 164 | ŋ   | 180 |  | 196 | ℐ | 212 |  |  | 228 | Σ | 244 | ≫ |
| 133 | å   | 149 | ò   | 165 | Ñ   | 181 |  | 197 | ℐ | 213 |  |  | 229 | σ | 245 | Ｊ |
| 134 | Á   | 150 | Ù   | 166 | ą   | 182 |  | 198 | ℐ | 214 |  |  | 230 | μ | 246 | Ｚ |
| 135 | ç   | 151 | Ù   | 167 | ő   | 183 |  | 199 | ℐ | 215 |  |  | 231 | τ | 247 | Ｚ |
| 136 | ë   | 152 | Ź   | 168 | ɕ   | 184 |  | 200 | ℐ | 216 |  |  | 232 | Φ | 248 | Ｚ |
| 137 | Ë   | 153 | Ô   | 169 | Ł   | 185 |  | 201 | ℐ | 217 |  |  | 233 | Ω | 249 | Ｚ |
| 138 | Ë   | 154 | Ū   | 170 | ń   | 186 |  | 202 | ℐ | 218 |  |  | 234 | Ø | 250 | Ｚ |
| 139 | Í   | 155 | Ć   | 171 | ½   | 187 |  | 203 | ℐ | 219 |  |  | 235 | ø | 251 | ≫ |
| 140 | Í   | 156 | Ė   | 172 | ¼   | 188 |  | 204 | ℐ | 220 |  |  | 236 | ∞ | 252 | ≫ |
| 141 | í   | 157 | Ő   | 173 | I   | 189 |  | 205 | ℐ | 221 |  |  | 237 | φ | 253 | ≫ |
| 142 | Å   | 158 | ≈   | 174 | «   | 190 |  | 206 | ℐ | 222 |  |  | 238 | ø | 254 | Ｚ |
| 143 | Å   | 159 | f   | 175 | »   | 191 |  | 207 | ℐ | 223 |  |  | 239 | Φ | 255 | Ｚ |

Source: www.LookupTables.com
TEXT Editors

**vi**
- Available on all UNIX-like systems (Linux included), i.e., also on lab workstations (type `vi` or `vi file_name`)
- Free Windows implementation available (once you learn vi, you can just use one editor everywhere)
- Runs locally on Linux machine (no network transfers)
- User interface rather peculiar (no nice buttons to click, need to remember quite a few keyboard commands instead)
- Some love it, some hate it

**gvim**
- Vi (see above) with a graphical interface – X-Windows needed. Windows version available.

**nano**
- Available on most Linux machines (our workstations included; type `nano` or `nano file_name`)
- Intuitive user interface. Keyboard commands-driven, but help always displayed on bottom bar (unlike in vi).
- Runs locally on Linux machine (no network transfers during editing)
TEXT Editors

**gedit** (installed on lab workstations; just type `gedit` or `gedit file_name` to invoke)
- X-windows application – need to have X-ming running on client PC.
- [May be slow on slow networks...](#)

**edit+** ([http://www.editplus.com/](http://www.editplus.com/))
- Commercial product
- Runs on a local machine (laptop) and transfers data to/from Linux workstation as needed
- Can browse Linux directories in a Windows-like file explorer
- [May be slow on slow networks](#)
- Some people swear by it

**emacs** (installed on lab workstations)

**Xcode** (Mac)

**Notepad** (Windows)
TEXT Files on Unix, Windows and Mac

End-of-line problem:

- Unix: \n CR 10 0x0a
- Windows \n\r CR+LF 10 13 0x0a 0x0d
- Mac (old) \r LF 13 0x0d
- Mac (new) \n CR 10 0x0a

Make sure files transferred from one system to another are properly converted.

On Linux there is a set of nice utilities:

- `unix2dos file_name`
- `dos2unix file_name`
- `unix2mac file_name`
- `mac2unix file_name`

Example: Windows and Unix files on Windows
Vi basics

Opening a file:

vi my_reads.fastq (open the file my_reads.fastq in the current directory for editing; if the file does not exist, it will be created)

Command mode: typing will issue commands to the editor (rather than change text itself)
Edit mode: typing will enter/change text in the document

<Esc> exit edit mode and enter command mode (this is the most important key – use it whenever you are lost)

The following commands will take you to edit mode:

i enter insert mode
r single replace
R multiple replace
a move one character right and enter insert mode
o start a new line under current line
O start a new line above the current line

The following commands operate in command mode (hit <Esc> before using them)

x delete one character at cursor position
dd delete the current line
G go to end of file
1G go to beginning of file
154G go to line 154
$ go to end of line
1 go to beginning of line
:q! exit without saving
:w save (but not exit)
:wq! save and exit

Arrow keys: move cursor around (in both modes)
Look of a typical Perl script:

```perl
#!/usr/local/bin/perl

#this is my first Perl script

print "Hello, CBSU\n";
```
"shebang" notation – path to the program to interpret the script, must be the first line and start with #!

```perl
#!/usr/local/bin/perl
#this is my first Perl script
print "Hello, CBSU\n";
```

anything starting with # is a comment, unless it is #! in the first line

statement ends with a semicolon

function to print out text
“shebang” notation – path to the program to interpret the script, must be the first line and start with #!

```
#!/usr/local/bin/perl

#this is my first Perl script

print("Hello, CBSU\n");
```

- Anything starting with # is a comment, unless it is #! in the first line.
- Parentheses can be always omitted, unless it changes the meaning of expression.
- Statement ends with semicolon.
- Function to print out text.
Strings in Perl

• Sequence of characters – simple (ASCII) or extended (Unicode, wide)

• Special characters like NL or CR are represented as \xxxx (C notation)
  o \n     new line (NL)
  o \t     tab character
  o \r     return (CR)
  o \x0a any character represented by hex number (0a = 10 = NL)
  o \”     double quotation
  o \’     single quotation
  o \\     backslash

• Strings may be joined by ‘.’ operator

  “string 1” . “string 2”  <=>  “string 1 string 2”

• Some characters have special meaning in Perl, most prominently $ and @
  o \$ {dollar}
  o \@ {at}
Strings in Perl

• Single Quoted

Single quoted strings have LITERAL meaning – no special characters are recognized:

‘string 1’ string 1
‘string 1\n’ string 1{backslash}n
‘\’string 1\’’ ‘string 1’
‘ string 1\\1’ string 1\1

• Double-Quoted

Double quoted strings do interpret special characters properly:

“string 1\n” string 1{new line}
“\“string 1\““ “string 1“
Perl installation and usage depends on the OS

External Perl libraries (modules) are accessible via CPAN

CPAN = Comprehensive Perl Archive Network

You can download and use any of publicly available modules in your programs
Perl on Linux

• Almost always installed as a part of the system, if not ask your system admin

• Usually it is /usr/bin/perl or /usr/local/bin/perl

• May be several versions installed, each with its own libraries and features

• Version can be checked with command
  >perl -v
  >/usr/bin/perl -v

• If you need a particular Perl installation in your program, write it into the first line
  #!/usr/local/special/bin/perl

• If you need default Perl installation in your program, write it into the first line
  #!/usr/bin/env perl

• Once invoked, Perl interpreter knows where its system-wide modules reside
Perl on Linux

Execute Perl program

- If the scripts has executable right
  
  `./script_name.pl`

  `./script_name.pl >& output`

- Regardless of executable right
  
  `perl script_name`

- Compile (verify) Perl program
  
  `perl -c script_name`

Make script executable:

`chmod u+x script_name`
Perl on Linux

If you need custom modules located in a custom place:

• write it into first line

    #!/usr/local/bin/perl -I /home/jarekp/my_modules

• set environmental variable

    PERL5LIB=/home/jarekp/my_modules:/usr/another/path/lib; export PERL5LIB

• Execute explicitly with Perl interpreter and options

    >perl -I /home/jarekp/my_modules my_script.pl
#!/usr/local/bin/perl

#this is my first Perl script

print "Hello, CBSU\n";
Perl on Linux: CPAN

Two interfaces to CPAN

> cpan

> perl -MCPAN -e shell

Then you can type command

install modname
r modname
upgrade modname
m modname

- install module modname
- report if upgrade is available
- upgrade
- info about modname

Remember: there is a cpan for EACH Perl installation, make sure you are using the right one
Perl on Linux: CPAN

Listing installed modules is not easy by itself, but there is a shortcut

```bash
>instmodsh
```

```
I – list all

(instmodsh may take a LONG time to load, be patient)
```

Remember – there is a version of instmodsh for EACH Perl installed:

```
>/usr/bin/instmodsh

>/usr/local/bin/instmodsh
```
Perl on Linux: CPAN

If you want to install a module for your own use, without being an admin:

Configure cpan (only first time)

> cpan
  o conf makepl_arg INSTALL_BASE=~/%myPERL_LIB
  o conf mbuild_arg INSTALL_BASE=~/%myPERL_LIB
  o conf prefs_dir ~/ myPERL_LIB/prefs
  o conf commit

Install module(s)

> cpan
install modname

Set up environment so Perl knows where to look

PERL5LIB=/home/jarekp/myPERL_LIB/lib/perl5:$PERL5LIB
Export PERL5LIB

Need to reset CPAN:

o conf init
Perl on Windows

Recommended Perl is ActivePerl: [http://www.activestate.com/activeperl](http://www.activestate.com/activeperl)

Download binary and install – choose free version.

“shebang” line of any script is ignored on Windows

Windows recognizes Perl scripts by extension .pl

There is a nice GUI to CPAN

Example of script and GUI
Perl on Mac

Similarly as on Linux it comes preinstalled on OS X.

All Linux information should apply.
A bit more complicated script

#!/usr/local/bin/perl

use warnings;
use Bio::Perl;

#this is my first Perl script

print “Hello, CBSU\n”;}
use ModuleName;
Declares usage of Perl module “ModuleName”, includes all proper definitions

use warnings;
Declares use of “warnings” module – Perl will now report any place it thinks is ambiguous or suspicious: same as >perl –w

use Bio::Perl;
Declares use of BioPerl module – more details later
"use" statement can be declared as a parameter of Perl interpreter

```
>perl -MBio::Perl
```

... and then something can be executed ...

```
>perl -MBio::Perl -e "print \"OK\n\";"
```

If Bio::Perl is installed it will print "OK", otherwise an error will occur.

Easy way to check if a module is installed.

**Example:** CPAN installation of Template::HTML
Exercises

1. Write a Perl program that prints your name and e-mail in the following format in one line:
   first_name last_name <emailaddr@domain.edu>

2. Are the following modules installed on your BioHPC Lab machine?

   Net::Ping
   XML::Special
   Net::Telnet
   CBSU::HDF5