1 Administrivia

Announcements

Assignment

None.

From Last Time

Logical and branch instructions.

Outline

1. Compiling HLL control structures.

2. Class assignment.

Coming Up

Intro to Linux.
2 Compiling HLL Control Structures

Write MIPS code fragments corresponding to the following:

1. Compiling an if:

<table>
<thead>
<tr>
<th>HLL Code</th>
<th>Assembly Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Condition</td>
<td>Conditional branch on</td>
</tr>
<tr>
<td>If block</td>
<td>!Condition to Else label</td>
</tr>
<tr>
<td>Else block</td>
<td>Branch to EndIf label</td>
</tr>
<tr>
<td>Next instruction</td>
<td>Else:</td>
</tr>
<tr>
<td></td>
<td>EndIf:</td>
</tr>
</tbody>
</table>

```cpp
def main()
    int i = 1;
    if (i < 12)
        ++i;
    else
        --j;
```

2. Compiling a loop:

<table>
<thead>
<tr>
<th>HLL Code</th>
<th>Assembly Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Condition</td>
<td>Conditional branch on</td>
</tr>
<tr>
<td>Loop block</td>
<td>!Condition to EndLoop label</td>
</tr>
<tr>
<td>Next instruction</td>
<td>BeginLoop:</td>
</tr>
<tr>
<td></td>
<td>Loop block</td>
</tr>
<tr>
<td></td>
<td>Branch to BeginLoop label</td>
</tr>
<tr>
<td></td>
<td>EndLoop:</td>
</tr>
<tr>
<td></td>
<td>Next instruction</td>
</tr>
</tbody>
</table>

```c
for (int i = 1; i < 12; ++i)
    i = i + 1;
```
j = 0;
while (i < 200)
{
    j += i;
    i *= i;
}

3 Class Assignment

Write MIPS code corresponding/solving each of the following:

1. j = 0;
   for (i = 0; i < 10; ++i)
       j += i;

2. j = 0;
   for (i = 0; i < 10; ++i)
       if (i > 5)
           j += i;

3. while (i > 0 && i < 10)
   ++i;

4. if (i < 12 && j > 3 || k != 0)
   ++i;
   else if (i == 33)
       --j;
   else
       k += 2;

5. (3.9 from the text) The naive way of compiling

   while (save[i] == k)
       i += k;

requires execution of both a conditional branch and an unconditional jump each time through the loop. Produce the naive code.

Optimize the naive code so that only a conditional branch is executed each time through the loop.
6. (3.24 from the text, a variation) Write a code segment which takes two “parameters:”

   (a) An ASCII character in $a0$.

   (b) A pointer to a NULL-terminated string in $a1$.

and “returns” a count of the number of occurrences of the character in the string in $v0$. 