Princeton University
COS 217: Introduction to Programming Systems
Spring 2005 Final Exam Answers

Question 1

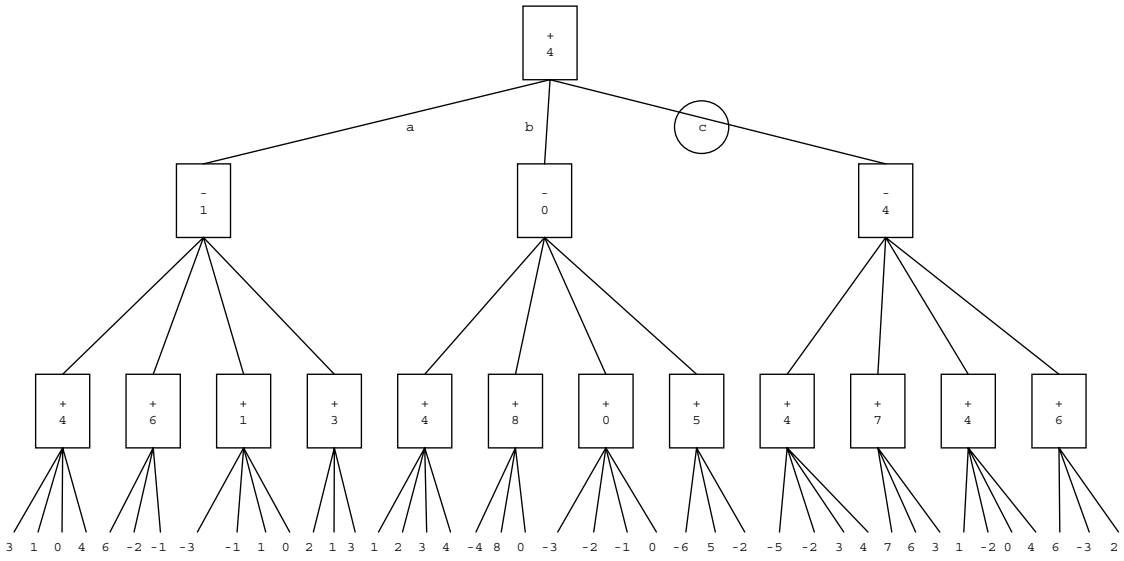
The heuristic function will be evaluated 12 times.

Explanation: There are 12 game states at depth 2.

Question 2

The heuristic function will be evaluated 42 times. MAX will choose move c.

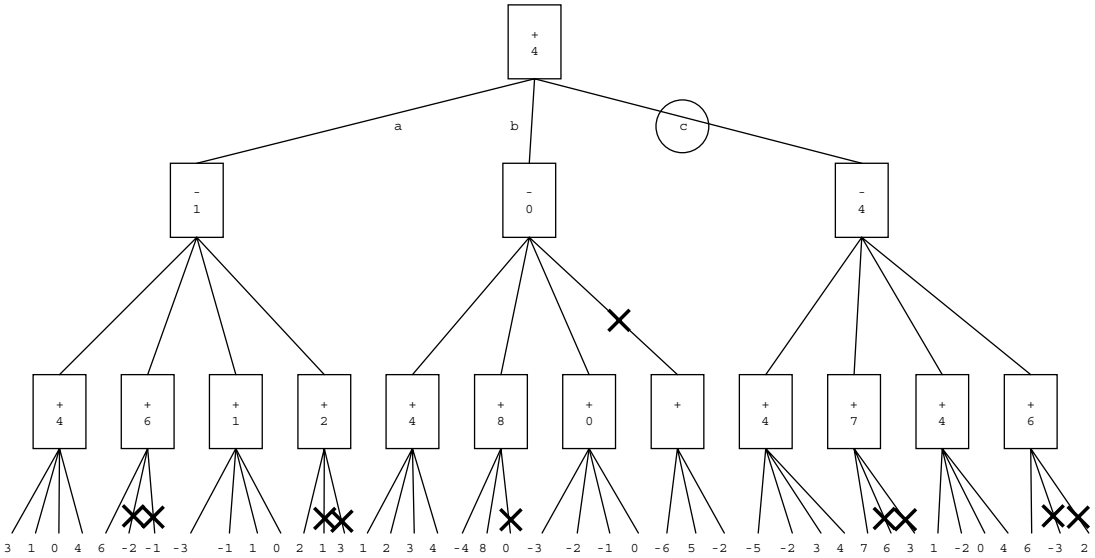
Explanation: There are 42 game states at depth 3.



Question 3

The heuristic function will be evaluated 30 times. MAX will choose move c.

Explanation:



Question 4

MAX's best move would be b.

Explanation: MAX chooses b. MAX then convinces MIN to choose the second child at level 2. MAX then chooses the second child at level 3. The heuristic value of that child is 8. That is the maximum heuristic value of all game states at level 3.

Question 5

Assume that the linked list resides in the heap at these pretend addresses:

Pretend Address	Contents
2000	5
2004	2008
2008	2
2012	2016
2016	6
2020	NULL

Then this is a picture of the stack:

Offset vs. EBP	Pretend Address	Contents	Meaning
f stack frame:			
-12	1000	33	sum
-8	1004	???	copy.val
-4	1008	???	copy.next
0	1012	1036	old value of EBP
f stack frame:			
4	1016	???	return address
8	1020	NULL	p->next (becomes p)
12	1024	1028	&copy (becomes r)
	1028	6	copy.val
	1032	1052	copy.next
	1036	1060	old value of EBP
f stack frame:			
	1040	???	return address
	1044	2016	p->next (becomes p)
	1048	1052	&copy (becomes r)
	1052	2	copy.val
	1056	1076	copy.next
	1060	1084	old value of EBP
f stack frame:			
	1064	???	return address
	1068	2008	p->next (becomes p)
	1072	1076	&copy (becomes r)
	1076	5	copy.val
	1080	NULL	copy.next
	1084	1104	old value of EBP
main stack frame:			
	1088	???	return address
	1092	2000	mylist (becomes p)
	1096	NULL	(becomes r)
	1100	2000	mylist
	1104	???	old value of EBP
_start stack frame:			
	1108	???	return address
	1112	???	argc
	1116	???	argv

Question 6

```
The program prints 33.

Explanation:

2 * 0 + 6 = 6
2 * 6 + 2 = 14
2 * 14 + 5 = 33
```

Question 7

```
        ## ============================================================
        ## Spring 2005 Final Exam Question 7
        ## ============================================================

        ## ============================================================
                              .section ".text"
        ## ============================================================

        ## ------------------------------------------------------------
        ## int f(struct list *p, struct list *r)
        ##
        ## Local variable offsets:
                .equ SUM,       -12
                .equ COPY,       -8
                .equ COPY_VAL,   COPY
                .equ COPY_NEXT,  COPY+4
        ## Formal parameter offsets:
                .equ P,           8
                .equ R,          12
        ## ------------------------------------------------------------

        .globl  f
        .type   f,@function

f:

        pushl   %ebp
        movl    %esp, %ebp

        ## struct list copy;
        subl    $8, %esp

        ## if (p != NULL) goto else1;
        cmpl    $0, P(%ebp)
        jne     else1

        ## int sum = 0;
        pushl   $0

beginfor1:

        ## if (r == NULL) goto endfor1;
        cmpl    $0, R(%ebp)
        je      endfor1

        ## sum = 2*sum + r->val;
        movl    SUM(%ebp), %eax
        sall    $1, %eax
        movl    R(%ebp), %ecx
        addl    (%ecx), %eax
        movl    %eax, SUM(%ebp)

        ## r = r->next;
        movl    R(%ebp), %eax
        movl    4(%eax), %eax
```

```
        movl    %eax, R(%ebp)

        ## goto beginfor1;
        jmp     beginfor1

endfor1:

        ## return sum;
        movl    SUM(%ebp), %eax
        movl    %ebp, %esp
        popl    %ebp
        ret

else1:

        ## copy.val = p->val;
        movl    P(%ebp), %eax
        movl    (%eax), %eax
        movl    %eax, COPY_VAL(%ebp)

        ## copy.next = r;
        movl    R(%ebp), %eax
        movl    %eax, COPY_NEXT(%ebp)

        ## return f(p->next, &copy);
        leal    COPY(%ebp), %eax
        pushl   %eax
        movl    P(%ebp), %ecx
        pushl   4(%ecx)
        call    f
        addl    $8, %esp
        movl    %ebp, %esp
        popl    %ebp
        ret
```

Question 8

```
Declaration:

        void SymTable_mapWords(SymTable_T oSymTable,
           void (*pfApply)(const char *pcKey, void *pvValue, void *pvExtra),
           const void *pvExtra);

Definitions:

        struct MapWordsPair
        {
           const void *pvExtra;
           void (*pfApply)(const char *pcKey, void *pvValue, void *pvExtra);
        };

        static void SymTable_mapWordsHelp(const char *pcKey, void *pvValue, void *pvExtra)
        {
           struct MapWordsPair *psMapWordsPair = (struct MapWordsPair*)pvExtra;
           if (! isalpha(*pcKey))
              return;
           (*(psMapWordsPair->pfApply))(pcKey, pvValue, (void*)psMapWordsPair->pvExtra);
        }

        void SymTable_mapWords(SymTable_T oSymTable,
           void (*pfApply)(const char *pcKey, void *pvValue, void *pvExtra),
           const void *pvExtra)
        {
           struct MapWordsPair sMapWordsPair;
           sMapWordsPair.pvExtra = pvExtra;
           sMapWordsPair.pfApply = pfApply;
           SymTable_map(oSymTable, SymTable_mapWordsHelp, &sMapWordsPair);
        }
```

Question 9

$$\frac{p_0}{p_1} = \frac{P(v_0)\cdot\prod_{i\in test}P(a_i\mid v_0)\cdot\prod_{i\notin test}1-P(a_i\mid v_0)}{P(v_1)\cdot\prod_{i\in test}P(a_i\mid v_1)\cdot\prod_{i\notin test}1-P(a_i\mid v_1)}$$

$$= \frac{P(v_0)\cdot\prod_{i\in test}P(a_i\mid v_0)\cdot\prod_{i\notin test}1-P(a_i\mid v_0)\cdot\prod_{i\in test}1-P(a_i\mid v_0)\cdot\prod_{i\in test}1-P(a_i\mid v_1)}{P(v_1)\cdot\prod_{i\in test}P(a_i\mid v_1)\cdot\prod_{i\notin test}1-P(a_i\mid v_1)\cdot\prod_{i\in test}1-P(a_i\mid v_0)\cdot\prod_{i\in test}1-P(a_i\mid v_1)}$$

$$= \frac{P(v_0)\cdot\prod_{i\in test}P(a_i\mid v_0)\cdot\prod_{i\in test}1-P(a_i\mid v_1)\cdot\prod_{i\in test}1-P(a_i\mid v_0)\cdot\prod_{i\notin test}1-P(a_i\mid v_0)}{P(v_1)\cdot\prod_{i\in test}P(a_i\mid v_1)\cdot\prod_{i\in test}1-P(a_i\mid v_0)\cdot\prod_{i\in test}1-P(a_i\mid v_1)\cdot\prod_{i\notin test}1-P(a_i\mid v_1)}$$

$$= \frac{P(v_0)\cdot\prod_{i\in test}P(a_i\mid v_0)\cdot\prod_{i\in test}1-P(a_i\mid v_1)\cdot\prod_{i}1-P(a_i\mid v_0)}{P(v_1)\cdot\prod_{i\in test}P(a_i\mid v_1)\cdot\prod_{i\in test}1-P(a_i\mid v_0)\cdot\prod_{i}1-P(a_i\mid v_1)}$$

$$= \frac{P(v_0)\cdot\prod_{i}1-P(a_i\mid v_0)\cdot\prod_{i\in test}P(a_i\mid v_0)\cdot\prod_{i\in test}1-P(a_i\mid v_1)}{P(v_1)\cdot\prod_{i}1-P(a_i\mid v_1)\cdot\prod_{i\in test}P(a_i\mid v_1)\cdot\prod_{i\in test}1-P(a_i\mid v_0)}$$

$$= \frac{P(v_0)\cdot\prod_{i}1-P(a_i\mid v_0)\cdot\prod_{i\in test}P(a_i\mid v_0)\cdot(1-P(a_i\mid v_1)}{P(v_1)\cdot\prod_{i}1-P(a_i\mid v_1)\cdot\prod_{i\in test}P(a_i\mid v_1)\cdot(1-P(a_i\mid v_0)}$$

$$= \frac{P(v_0)}{P(v_1)}\cdot\prod_{i}\frac{1-P(a_i\mid v_0)}{1-P(a_i\mid v_1)}\cdot\prod_{i\in test}\frac{P(a_i\mid v_0)\cdot(1-P(a_i\mid v_1))}{P(a_i\mid v_1)\cdot(1-P(a_i\mid v_0))}$$

$$= A\cdot\prod_{i\in test}H_i$$

Question 10

Let S_i be hit score i. Furthermore, let S_i equal $\log(1/H_i)$. Then:

$$S_i = \log\left(\frac{1}{H_i}\right)$$

$$\exp(S_i) = \frac{1}{H_i}$$

$$H_i = \frac{1}{\exp(S_i)}$$

$$H_i = \exp(-S_i)$$

From the previous question:

$$pSpam = \frac{1}{\frac{p_0}{p_1} + 1}$$

$$pSpam = \frac{1}{A \cdot \prod_{i \in test} H_i + 1}$$

$$pSpam = \frac{1}{A \cdot \prod_{i \in test} \exp(-S_i) + 1}$$

$$pSpam = \frac{1}{A \cdot \exp\left(\sum_{i \in test} -S_i\right) + 1}$$

$$pSpam = \frac{1}{A \cdot \exp\left(-\sum_{i \in test} S_i\right) + 1}$$

Note that *pSpam* increases as the sum of the hit scores increases. More generally, the choice of $S_i = \log(1/H_i)$ makes *pSpam* a function of the sum of the hit scores, and vice-versa.

The Spam Assassin employees can use that equation to set the hit score of each feature, based upon the effect that they want the presence of that feature to have upon *pSpam*.

Question 11

You should set your threshold by using the equation from the previous answer. From the previous answer:

$$pSpam = \frac{1}{A \cdot \exp\left(- \sum_{i \in test} S_i \right) + 1}$$

You want to know the threshold (that is the sum of all S_i) that will make *pSpam* equal .9. Let's call the threshold t. So you should solve this equation for t:

$$.9 = \frac{1}{A \cdot \exp(-t) + 1}$$

[It was not necessary to solve the equation. Nevertheless, here's the solution...]

$$.9 = \frac{1}{A \cdot \exp(-t) + 1}$$

$$.9 \cdot \left(A \cdot \exp(-t) + 1 \right) = 1$$

$$.9 \cdot A \cdot \exp(-t) + .9 = 1$$

$$.9 \cdot A \cdot \exp(-t) = .1$$

$$A \cdot \exp(-t) = \frac{1}{9}$$

$$\exp(-t) = \frac{1}{9A}$$

$$-t = \log\left(\frac{1}{9A} \right)$$

$$t = -\log\left(\frac{1}{9A} \right)$$

Question 12

You could adjust $P(v_0)$ and $P(v_1)$, that is, the a priori probabilities that a message is ham or spam, respectively. Setting $P(v_0)$ lower and $P(v_1)$ higher would filter out more of the spam.

Question 13

We should not "brightly anticipate" that solution to the spam problem.

Apparently an entity named Habeas offers a "Sender Warranted Email" service. E-mail senders can purchase a license with Habeas. In return, the sender has the right to include the X-Habeas-SWE-n headers in his/her e-mail. Spam Assassin then considers the presence of such headers as evidence that the e-mail is not spam. In particular, as shown in the X-Spam-Report header, Spam Assassin assigns a -8 hit score to the presence of a Habeas warrant mark, thus substantially reducing the probability that it will classify the message as spam.

However... Clearly any e-mail sender can forge X-Habeas-SWE-n headers in a message, as indicated by the fact that the given message is spam, and yet contains the headers.

To combat such forgery, if someone receives an e-mail that is spam and contains the X-Habeas_SWE-n headers, he/she is supposed to report the forgery to http://www.habeas.com/report/.

However... How many e-mail recipients will have the knowledge and time to read spam message headers and lodge such reports? And couldn't spam generators easily flood Habeas with bogus reports?

So, in summary, the feature can be helpful. But we should not "brightly anticipate" that it will be a "solution" to the spam problem.