UCLA

UNIVERSITY OF CALIFORNIA, LOS ANGELES

BERKELEY • DAVIS • IRVINE • LOS ANGELES • RIVERSIDE • SAN DIEGO • SAN FRANCISCO

SANTA BARBARA • SANTA CRUZ

CS 33
Midterm

All answers must be written on the answer sheet (last page of the exam).

All work should be written directly on the exam, use the backs of pages if needed.

This is an open book, open notes quiz – but you cannot share books or notes. An ASCII table is on the second to last page if you need it.

I will follow the guidelines of the university in reporting academic misconduct – please do not cheat.

NAME: Kevin Zhang

ID: _____

Problem 1: 28

Problem 2: 15

Problem 3: 20

Problem 4: 42

Total: 105

− TMIN − TMax

1. ***C If You Can Solve This (28 points)***: The following problem assumes the following declarations:

```
int x = rand();
int y = rand();
int z = rand();
unsigned ux = (unsigned) x;
unsigned uy = (unsigned) y;
```

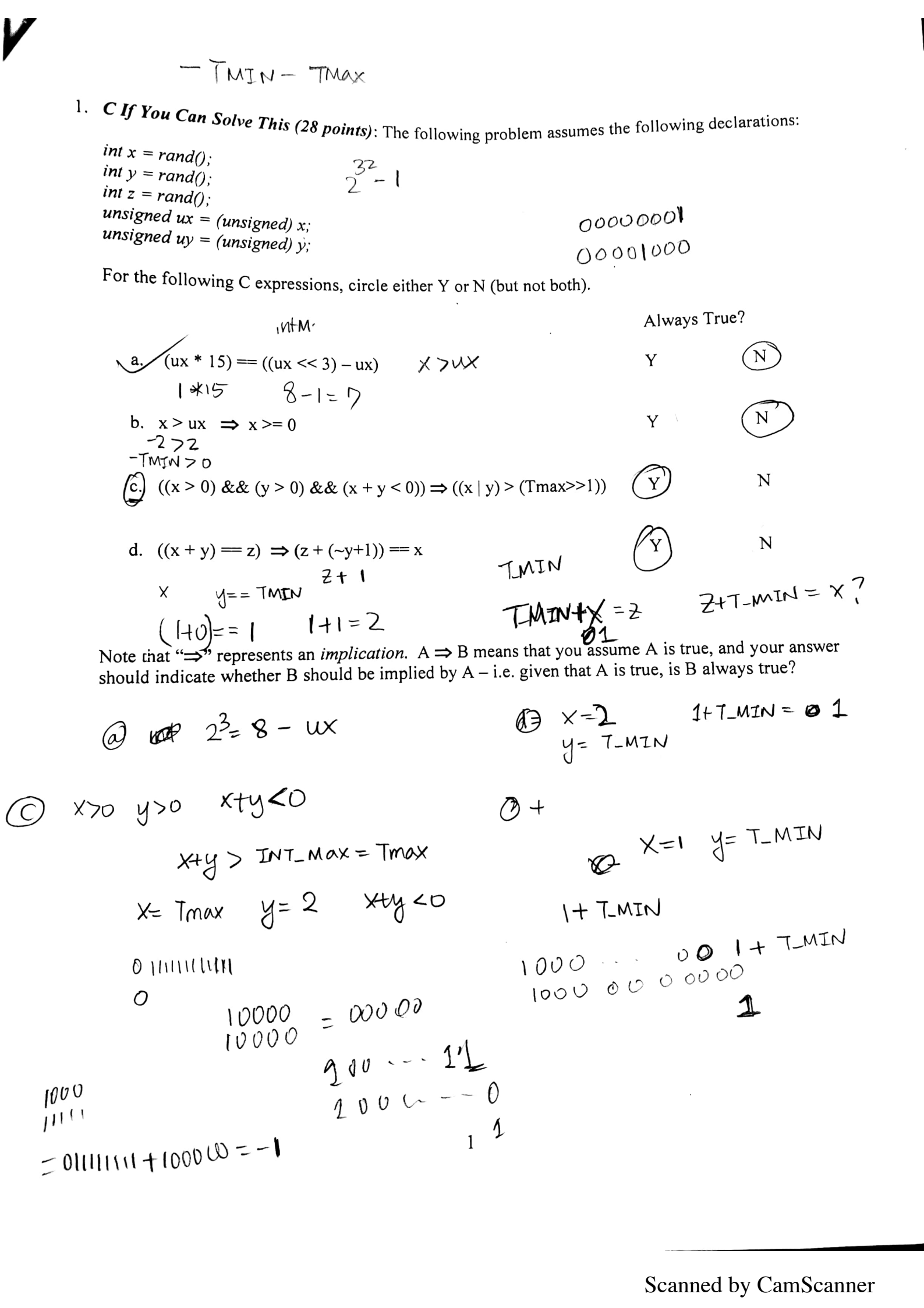
$2^{32}-1$

00000001
00001000

For the following C expressions, circle either Y or N (but not both).

	Always True?	
a. (ux * 15) == ((ux << 3) − ux)	Y	(N)
b. x > ux ⇒ x >= 0	Y	(N)
c. ((x > 0) && (y > 0) && (x + y < 0)) ⇒ ((x | y) > (Tmax>>1))	(Y)	N
d. ((x + y) == z) ⇒ (z + (~y+1)) == x	(Y)	N

Note that "⇒" represents an *implication*. A ⇒ B means that you assume A is true, and your answer should indicate whether B should be implied by A – i.e. given that A is true, is B always true?



2. ***Complete Dis-Array (10 points)***: Consider the following array declaration:

```
char * array4[7][7];
```

How much space (in bytes) would this array declaration allocate on an x86-64 machine?

49 * 8 = 392

3. ***Jump Table Blues (20 points)***: A C function includes this switch statement:

```
switch(a){

 case 100:
   i++;
   j++;
   break;

 case 102:
   i+=2;
   j+=2;
   break;

 case 103:
   i++;
 case 104:
   j+=2;
   break;

 case 105:
   i+=2;
   j++;
   break;

 default:
   i=0;
   j=0;
}
```

Some of the assembly code (x86-64) for this function is below:

```
0000000000400544 <func0>:
  400544: 55                    push   %rbp
  400545: 48 89 e5              mov    %rsp,%rbp
  400548: 48 83 ec 10           sub    $0x10,%rsp
  40054c: 89 7d fc              mov    %edi,-0x4(%rbp)
  40054f: 89 75 f8              mov    %esi,-0x8(%rbp)
  400552: 89 55 f4              mov    %edx,-0xc(%rbp)
  400555: 8b 45 f4              mov    -0xc(%rbp),%eax
```

1st arg
2nd arg
3rd arg
3rd arg

3rd arg - 0x64

```
400558: 83 e8 64                        sub     $0x64,%eax
40055b: 83 f8 05                        cmp     $0x5,%eax
40055e: 77 34                           ja      400594 <func0+0x50>
400560: 89 c0                           mov     %eax,%eax
400562: 48 8b 04 c5 28 08 40            mov     0x400828(,%rax,8),%rax
400569: 00
40056a: ff e0                           jmpq    *%rax
```

cant be above 105

(3rd arg - 64) × 8 + 0x400828

Which of the following memory dumps could possibly include the jump table for this switch statement:

Option **Brown**:

```
(gdb) x/64bx 0x400828
0x400828: 0x6c 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400830: 0x94 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400838: 0x76 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400840: 0x80 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400848: 0x84 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400850: 0x8a 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400858: 0x62 0x72 0x6f 0x77 0x6e 0x0a 0x00 0x70
0x400860: 0x69 0x6e 0x6b 0x0a 0x00 0x72 0x65 0x64
```

case 100 → 0x400828
~~case 101~~ → 0x400830
case 102 → 0x400838
case 103 → 0x400840
case 104 → 0x400848
case 105 → 0x400850

40 05 6c
40 05 76
40 05 80
40 05 84
40 05 8a

Option **Tan**:

```
(gdb) x/64bx 0x400828
0x400828: 0x6c 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400830: 0x76 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400838: 0x94 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400840: 0x80 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400848: 0x84 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400850: 0x8a 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400858: 0x62 0x72 0x6f 0x77 0x6e 0x0a 0x00 0x70
0x400860: 0x69 0x6e 0x6b 0x0a 0x00 0x72 0x65 0x64
```

100 → 0x400828
101 → 0x400830
102 → 0x400838
103 → 0x400840
104 → 0x400848
105 → 0x400850

0x64

6C) 10
76) 10
80) 4
84) 16
8a) 10
94

Option **Pink**:

```
(gdb) x/64bx 0x400828
0x400828: 0x6c 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400830: 0x94 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400838: 0x76 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400840: 0x80 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400848: 0x84 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400850: 0x94 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400858: 0x8a 0x05 0x40 0x00 0x00 0x00 0x00 0x00
0x400860: 0x62 0x72 0x6f 0x77 0x6e 0x0a 0x00 0x70
```

84 doesn't belong

Answer *ONE* of **Brown**, **Tan**, or **Pink** on the answer sheet.

4. ***This Problem is a Pain in My Big Endian (42 points):***

Consider the following C code:

```
struct node_t {
  short key;
  long val;
  char label[10];
  struct node_t * next;
};

struct node_t * a[4];

int hash(int key)
{
  return key%4;
}

void search(int key)
{
  int i=hash(key);
  struct node_t *ptr;
  int j=0;
  ptr=a[i];
  while (ptr)
    {
      if (ptr->key==key)
      printf("SEARCH FOUND: %s\n", ptr->label);
      ptr=ptr->next;
    }
}
```

You need to figure out what is printed out when the following function call is performed:

search(88);

The next four pages contain everything you need to solve this problem. ***Fill in all blanks*** on the answer sheet for credit.

Here is some gdb interaction (on an x86-64 machine) which should prove useful – a breakpoint is set after the invocation of the function search:

```
(gdb) break *0x400a07
Breakpoint 1 at 0x400a07
(gdb) run
Starting program

Breakpoint 1, 0x0000000000400a07 in main ()
(gdb) print &a
$1 = (struct node_t *(*)[4]) 0x601320

(gdb) x/8192bx 0x601320
0x601320: 0x10 0x23 0x60 0x00 0x00 0x00 0x00 0x00
0x601328: 0x40 0x23 0x60 0x00 0x00 0x00 0x00 0x00
0x601330: 0x70 0x23 0x60 0x00 0x00 0x00 0x00 0x00
0x601338: 0xa0 0x23 0x60 0x00 0x00 0x00 0x00 0x00
                ... skipping ahead ...
0x601fe0: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x601fe8: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x601ff0: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x601ff8: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602000: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602008: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602010: 0x16 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602018: 0x1c 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602020: 0x77 0x6f 0x6c 0x66 0x00 0x00 0x00 0x00
0x602028: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602030: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602038: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602040: 0x37 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602048: 0x2b 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602050: 0x62 0x65 0x61 0x72 0x00 0x00 0x00 0x00
0x602058: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602060: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602068: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602070: 0x21 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602078: 0x48 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602080: 0x74 0x69 0x67 0x65 0x72 0x00 0x00 0x00
0x602088: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602090: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602098: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6020a0: 0x0b 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6020a8: 0x4f 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6020b0: 0x6c 0x69 0x6f 0x6e 0x00 0x00 0x00 0x00
0x6020b8: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6020c0: 0x40 0x20 0x60 0x00 0x00 0x00 0x00 0x00
```

602310

602340

602370

6023a0

node * a[0]

```
0x6020c8: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6020d0: 0x2c 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6020d8: 0x17 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6020e0: 0x68 0x69 0x70 0x70 0x6f 0x00 0x00 0x00
0x6020e8: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6020f0: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6020f8: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602100: 0x42 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602108: 0x46 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602110: 0x7a 0x65 0x62 0x72 0x61 0x00 0x00 0x00
0x602118: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602120: 0x10 0x20 0x60 0x00 0x00 0x00 0x00 0x00
0x602128: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602130: 0x4d 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602138: 0x37 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602140: 0x63 0x61 0x72 0x69 0x62 0x6f 0x75 0x00
0x602148: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602150: 0x70 0x20 0x60 0x00 0x00 0x00 0x00 0x00
0x602158: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602160: 0x58 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602168: 0x27 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602170: 0x66 0x65 0x72 0x72 0x65 0x74 0x00 0x00
0x602178: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602180: 0xd0 0x20 0x60 0x00 0x00 0x00 0x00 0x00
0x602188: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602190: 0x63 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602198: 0x45 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6021a0: 0x6d 0x65 0x65 0x72 0x63 0x61 0x74 0x00
0x6021a8: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6021b0: 0xa0 0x20 0x60 0x00 0x00 0x00 0x00 0x00
0x6021b8: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6021c0: 0x34 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6021c8: 0x01 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6021d0: 0x77 0x6f 0x6d 0x62 0x61 0x74 0x00 0x00
0x6021d8: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6021e0: 0x60 0x21 0x60 0x00 0x00 0x00 0x00 0x00
0x6021e8: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6021f0: 0x3d 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6021f8: 0x29 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602200: 0x67 0x6f 0x72 0x69 0x6c 0x6c 0x61 0x00
0x602208: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602210: 0x30 0x21 0x60 0x00 0x00 0x00 0x00 0x00
0x602218: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602220: 0x4a 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602228: 0x28 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602230: 0x67 0x69 0x72 0x61 0x66 0x66 0x65 0x00
0x602238: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

NULL

66 65 72 72 65

node 4

60 20 d0

node 3

602160

58 in hex

note next 1 →

6021C0

6022 80

note a[6] →

```
0x602240: 0x00 0x21 0x60 0x00 0x00 0x00 0x00 0x00
0x602248: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602250: 0x55 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602258: 0x05 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602260: 0x62 0x75 0x66 0x66 0x61 0x6c 0x6f 0x00
0x602268: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602270: 0xf0 0x21 0x60 0x00 0x00 0x00 0x00 0x00
0x602278: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602280: 0x20 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602288: 0x19 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602290: 0x62 0x69 0x73 0x6f 0x6e 0x00 0x00 0x00
0x602298: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6022a0: 0xc0 0x21 0x60 0x00 0x00 0x00 0x00 0x00
0x6022a8: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6022b0: 0x2d 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6022b8: 0x5f 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6022c0: 0x76 0x75 0x6c 0x74 0x75 0x72 0x65 0x00
0x6022c8: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6022d0: 0x50 0x22 0x60 0x00 0x00 0x00 0x00 0x00
0x6022d8: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6022e0: 0x2f 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6022e8: 0x04 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6022f0: 0x65 0x61 0x67 0x6c 0x65 0x00 0x00 0x00
0x6022f8: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602300: 0x90 0x21 0x60 0x00 0x00 0x00 0x00 0x00
0x602308: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602310: 0x30 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602318: 0x2a 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602320: 0x66 0x61 0x6c 0x63 0x6f 0x6e 0x00 0x00
0x602328: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602330: 0x80 0x22 0x60 0x00 0x00 0x00 0x00 0x00
0x602338: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602340: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602348: 0x36 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602350: 0x68 0x61 0x77 0x6b 0x00 0x00 0x00 0x00
0x602358: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602360: 0xb0 0x22 0x60 0x00 0x00 0x00 0x00 0x00
0x602368: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602370: 0x32 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602378: 0x4f 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602380: 0x6f 0x77 0x6c 0x00 0x00 0x00 0x00 0x00
0x602388: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x602390: 0x20 0x22 0x60 0x00 0x00 0x00 0x00 0x00
0x602398: 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6023a0: 0x33 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6023a8: 0x37 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6023b0: 0x73 0x6e 0x61 0x6b 0x65 0x00 0x00 0x00
```

```
0x6023b8:  0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x6023c0:  0xe0 0x22 0x60 0x00 0x00 0x00 0x00 0x00
0x6023c8:  0x41 0x0c 0x02 0x00 0x00 0x00 0x00 0x00
0x6023d0:  0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

ASCII Table

Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-	63	3F	?	95	5F	_	127	7F	DEL

0x66 65 72 72 65 74

f e r r e t

Answer Sheet

Name: Kevin Zhang

1. Circle the correct responses:

Y (N)

Y (N)

(Y) N

(Y) N

2. Space (in bytes) for the array declaration: 392

3. Answer ***ONE*** of **`Brown`**, **`Tan`**, or **`Pink`**: Brown

4. Fill in all blanks below

The value of variable *i* in function search(): 0

The value of *a[i]* in function search(): 0x602310

The value of *ptr* at the time of execution of the printf statement in function search(): 0x602160

The value printed by the printf statement in function search(): SEARCH FOUND: ferret