PS6A

Graded

Student

Jae Hong Lee

Total Points

94 / 99 pts

Question 1

Question 1

1 / 1 pt

✔ **+ 1 pt** Correct

Question 2

Question 2

1 / 1 pt

✔ **+ 1 pt** Correct

Question 3

Question 3

1 / 1 pt

✔ **+ 1 pt** Correct

Question 4

Question 4

1 / 1 pt

✔ **+ 1 pt** Correct

Question 5

Question 5

1 / 1 pt

✔ **+ 1 pt** Correct

Question 6

Question 6

1 / 1 pt

✔ **+ 1 pt** Correct

Question 7

Question 7

1 / 1 pt

✔ **+ 1 pt** Correct

Question 8

Question 8

1 / 1 pt

✔ + 1 pt Correct

Question 9

Question 9

1 / 1 pt

✔ + 1 pt Correct

Question 10

Question 10

1 / 1 pt

✔ + 1 pt Correct

Question 11

Question 11 **14** / 14 pts

11.1 **Question 11a** **1** / 1 pt

✔ **+ 1 pt** Correct

11.2 **Question 11b** **1** / 1 pt

✔ **+ 1 pt** Correct

11.3 **Question 11c** **1** / 1 pt

✔ **+ 1 pt** Correct

11.4 **Question 11d** **1** / 1 pt

✔ **+ 1 pt** Correct

11.5 **Question 11e** **1** / 1 pt

✔ **+ 1 pt** Correct

11.6 **Question 11f** **1** / 1 pt

✔ **+ 1 pt** Correct

11.7 **Question 11g** **1** / 1 pt

✔ **+ 1 pt** Correct

11.8 **Question 11h** **1** / 1 pt

✔ **+ 1 pt** Correct

11.9 **Question 11i** **1** / 1 pt

✔ **+ 1 pt** Correct

11.10 **Question 11j** **1** / 1 pt

✔ **+ 1 pt** Correct

11.11 **Question 11k** **1** / 1 pt

✔ **+ 1 pt** Correct

11.12 **Question 11l** **1** / 1 pt

✔ **+ 1 pt** Correct

11.13 **Question 11m** **1** / 1 pt

✔ **+ 1 pt** Correct

11.14 **Question 11n** **1** / 1 pt

✔ **+ 1 pt** Correct

Question 12

Question 12 **5** / 5 pts

12.1 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

12.2 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

12.3 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

12.4 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

12.5 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

Question 13

Question 13 **6** / 6 pts

13.1 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

13.2 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

13.3 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

13.4 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

13.5 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

13.6 **(no title)** **1** / 1 pt

✔ **+ 1 pt** Correct

Question 14

Question 14 **4** / 4 pts

14.1 (no title) **2** / 2 pts

✔ **+ 2 pts** Correct

14.2 (no title) **2** / 2 pts

✔ **+ 2 pts** Correct

Question 15

Question 15 **6** / 8 pts

15.1 (no title) **2** / 2 pts

✔ **+ 2 pts** Correct

15.2 (no title) **2** / 2 pts

✔ **+ 2 pts** Correct

15.3 (no title) **0** / 2 pts

✔ **+ 0 pts** Incorrect: correct answer is 0x80000 0x80000

15.4 (no title) **2** / 2 pts

✔ **+ 2 pts** Correct

Question 16

Question 16 **37** / 40 pts

16.1 **(no title)** **14** / 15 pts

Incorrect

(1) 0x0000555555558010: Value: 'T' Name: bar[0]

(2) 0x0000555555558011: Value: 'o' Name: bar[1]

(3) 0x0000555555558012: Value: ' ' Name: bar[2]

(4) 0x0000555555558013: Value: 'C' Name: bar[3]

(5) 0x0000555555558014: Value: ' ' Name: bar[4]

(6) 0x0000555555558015: Value: 'o' Name: bar[5]

(7) 0x0000555555558016: Value: 'r' Name: bar[6]

(8) 0x0000555555558017: Value: ' ' Name: bar[7]

(9) 0x0000555555558018: Value: 'n' Name: bar[8]

(10) 0x0000555555558019: Value: 'o' Name: bar[9]

(11) 0x000055555555801a: Value: 't' Name: bar[10]

(12) 0x000055555555801b: Value: ' ' Name: bar[11]

(13) 0x000055555555801c: Value: 't' Name: bar[12]

(14) 0x000055555555801d: Value: 'o' Name: bar[13]

(15) 0x000055555555801e: Value: 0x00 Name: bar[14]

✔ **– 1 pt** (15) Incorrect

16.2 **(no title)** **3** / 4 pts

Partially Correct

(16) 0x000055555555801f: Value: 0x00 Name:

(17) 0x0000555555558020: Value: 0x0d Name: sv

(18) 0x0000555555558021: Value: 0x5e Name: sv

(19) 0x0000555555558022: Value: 0x00 Name:

✔ **– 0.5 pts** (16) Either value or name correct

✔ **– 0.5 pts** (19) Either value or name correct

16.3 **(no title)** **4.5** / 5 pts

Partially Correct

(20) 0x0000555555558023: Value: 0x00 Name:

(21) 0x0000555555558024: Value: 0x78 Name: iv

(22) 0x0000555555558025: Value: 0x56 Name: iv

(23) 0x0000555555558026: Value: 0x34 Name: iv

(24) 0x0000555555558027: Value: 0x12 Name: iv

✔ **– 0.5 pts** (20) Either value or name correct

16.4 **(no title)** **8** / 8 pts

✔ **– 0 pts** Correct

16.5 **(no title)** **7.5** / 8 pts

Partially Correct

(33) 0x0000555555558030: Value: 0x21 Name: p2

(34) 0x0000555555558031: Value: 0x80 Name: p2

(35) 0x0000555555558032: Value: 0x55 Name: p2

(36) 0x0000555555558033: Value: 0x55 Name: p2

(37) 0x0000555555558034: Value: 0x55 Name: p2

(38) 0x0000555555558035: Value: 0x55 Name: p2

(39) 0x0000555555558036: Value: 0x00 Name: p2

(40) 0x0000555555558037: Value: 0x00 Name: p2

✔ **– 0.5 pts** (33) Either value or name correct

Question 17

Question 17 **12** / 12 pts

17.1 **Box 1** **2** / 2 pts

✔ **+ 2 pts** Correct

17.2 **Box 2** **2** / 2 pts

✔ **+ 2 pts** Correct pointer in a

17.3 **Box 3** **4** / 4 pts

✔ **– 0 pts** Correct

17.4 **Box 4** **2** / 2 pts

✔ **– 0 pts** Correct

17.5 **Box 5** **2** / 2 pts

✔ **+ 2 pts** Correct

CS210 Fall 2023: PS6A

Instructions

For all multiple choice questions fill **ONE AND ONLY ONE circle**. Be sure to fill the circle in completely.
For all the questions we encourage you to login into the provided UNIX environment and explore your answers. For some questions, you must use the UNIX environment to answer them.

If you use checkmarks or other symbols the auto-grader may not be able to process your answer and will assign you a grade of zero.

All pages must have your name and id written on them. Unidentified pages will not be graded

First Name: Jae Hong Last Name: Lee

BU ID: U21565203

First Name: Jae Hong Last Name: Lee BU ID: U27562D3

Multiple Choice

1. (1 point) The C programming language
   - (●) is composed of two components: the Core Language and a standard Library of functions.
   - ( ) source is directly executable on the processor
   - ( ) allows one to do things not possible in assembly language
   - ( ) was developed to support direct manipulation of HTML
   - ( ) all of the above
   - ( ) none of the above

2. (1 point) The first stage of the C toolchain is
   - ( ) linking
   - (●) preprocessing
   - ( ) compilation
   - ( ) assembling
   - ( ) none of the above

3. (1 point) Each location of a process's address space
   - ( ) is valid and has an associated value
   - (●) represents a single byte of memory
   - ( ) has a fixed contents that never changes
   - ( ) is a variable sized array
   - ( ) all of the above
   - ( ) none of the above

4. (1 point) The indirection operator, in C,
   - (●) provides access to the value that a pointer is pointing to
   - ( ) provides the address of a value
   - ( ) provides the type of a value
   - ( ) provides the size of a value
   - ( ) all of the above
   - ( ) none of the above

5. (1 point) By the C calling conventions, if more than 6 arguments are passed into a C function, the remaining ones, that could not be assigned to registers, are pushed onto the stack
   - (●) True
   - ( ) False

First Name: Jue Hong Last Name: Lee BU ID: U27565203

6. (1 point) Structures in C
   - ◯ always requires at least 8 bytes
   - ◯ may not contain pointers to structures of other types
   - ◯ must be placed in the data section of the binary
   - ◯ must be allocated on the heap
   - ◯ all of the above
   - ● none of the above

7. (1 point) A function can be defined in many files
   - ◯ True
   - ● False

8. (1 point) What must be passed to `printf` to send bytes to standard output
   - ◯ a format specifier
   - ● a char pointer to a format string
   - ◯ a file descriptor of standard output
   - ◯ the size of bytes you want to send
   - ◯ all of the above
   - ◯ none of the above

9. (1 point) A local variable of a C function
   - ◯ must be placed on the stack
   - ◯ must be assigned to a register
   - ● will be placed on the stack or assigned to a register
   - ◯ will be placed in the heap if there are no available registers or space on the stack
   - ◯ none of the above

10. (1 point) How many times will the following loop execute?

```
int i=5, j=0;
while (i-j){}
```

   - ◯ 0
   - ◯ 1
   - ◯ 4
   - ◯ 5
   - ◯ 6
   - ● none of the above

First Name: Jae Hong Last Name: Lee BU ID: U27585203

11. Provide the following value of the C expressions as a 32-bit hexadecimal value. 8digits
DO NOT SKIP LEADING ZEROS:
Eg. a value of 0 should be written as `00000000` and 1 as `00000001`.

Assume a 64 bit computer that uses 2's complement representation, INT_MAX and INT_MIN are defined as the computer's signed 32-bit integer representation maximum and minimum value respectively, and:

int x = −1, y = 0x7eedface, z = INT_MIN, i = **sizeof** (**int**);

(a) (1 point) x : 0x FFFFFFFF

(b) (1 point) y : 0x 7eedface

(c) (1 point) z : 0x 80000000

(d) (1 point) i : 0x 00000004

(e) (1 point) z<<12 : 0x 00000000

(f) (1 point) z<<((i>>1)−1) : 0x 00000000

(g) (1 point) ~0 == (z + INT_MAX) : 0x 00000001

(h) (1 point) y & 0xffff : 0x 0000face

(i) (1 point) y>>16 : 0x 00007eed

(j) (1 point) (y>>16) | 0xffff : 0x 0000ffff

(k) (1 point) (~(0x10>>2)+1) == (x∗i) : 0x 00000001

(l) (1 point) (x+1) + −1 : 0x ffffffff

(m) (1 point) (~((~ x)<<1)) & y : 0x 7eedface

(n) (1 point) ((z+INT_MIN)<<i) ^ ((z+INT_MIN)<<i) : 0x 00000000

First Name: Jue Hong Last Name: Lee BU ID: U20565203

12. (5 points) Given the assembly code on the left fill in the blanks in the C code on the right that it corresponds to.

```
1       .intel_syntax noprefix
2       .text
3       .globl  func
4   func:
5       xor     eax, eax
6       xor     r8d, r8d
7   .L2:
8       add     r8d, DWORD PTR A[0+rax*4]
9       inc     rax
10      cmp     rax, 10
11      jne     .L2
12      mov     eax, r8d
13      ret
```

exx:0
r8d: 0 to E7
double
4 bytes
10 loop

```
1
2   #define B  10
3
4   #define T  int
5
6   T A[B];
7
8   T func()
9   {
10    int i;
11    T s = 0;
12
13    for ( i=0 ;
14
15          i < B;
16
17          i++ ) {
18
19
20      s = s+A[i];
21    }
22    return s;
23  }
```

First Name: Jae Hong Last Name: Lee BU ID: U25565203

13. (6 points) Given the assembly code on the left, fill in the blanks in the C code on the right that it corresponds to.

```
1   .intel_syntax noprefix
2   .text
3   .globl  func
4   func:
5       xor     rax, rax
6   .L2:
7       test    rdi, rdi
8       je      .L7
9       mov     rcx, QWORD PTR [rdi+16]
10      mov     rdx, QWORD PTR [rdi+24]
11      cmp     rsi, 12
12      jle     .L3
13      add     rax, QWORD PTR [rdi]
14      mov     rdi, rcx
15      jmp     .L2
16  .L3:
17      add     rax, QWORD PTR [rdi+8]
18      mov     rdi, rdx
19      jmp     .L2
20  .L7:
21      ret
```

rax: 0 + [rdi+8]
rdx:
rdi: 0
rcx:
8
16
struct
rax
4

rax: 0+[rdi+8] +

```
1   struct S {
2     long long x;
3     long long y;
4     struct S *f;
5     struct S *b;
6   };
7
8   long long
9   func(struct S *h,
10       long long v)
11  {
12    long long r = 0;
13
14    while (h != ________ ) {
15      if (v > ________ ) {
16
17        r = r + ________;
18
19        h = ________;
20      } else {
21
22        r = r + ________;
23
24        h = ________;
25      }
26    }
27    return r;
28  }
```

8 bytes
0
8
16
24
rax
0
12
h→x
h→f
h→y
h→b

First Name: Jae Hong Last Name: Lee BU ID: U20565203

14. (4 points) Given the assembly code on the left, fill in the blanks in the C code on the right that it corresponds to.

```
1       .intel_syntax noprefix
2       .text
3       .globl  f
4   f:
5       imul    rdi, rdi, 7
6       add     rdi, rsi
7       mov     rax, QWORD PTR A[0+rdi*8]
8       ret
9       .comm   A,280,32
```

name size Alignment

rdi = (rdi*7 + rsi) * 8

rax = (rdi*7 + rsi) * 8

280 = 8 * 7 * 5

40 56

5, 7

```
1
2   #define C __5_____
3
4   #define R __7_____
5
6   long long A[C][R];
7
8   long long f(long long c,
9               long long r)
10  {
11    return A[c][r];
12  }
```

First Name: Joe Hong Last Name: Lee BU ID: U27565203

15. Present the output for each line for the program below. Assume that it is compiled and executed on a 64-bit, little endian computer that uses 2's complement representation.

```
#include <stdio.h>
typedef unsigned char *byte_pointer;

void show_bytes(byte_pointer start, int len) {
  int i;
  for(i=0; i<len; i++)
    printf(" %.2x", start[i]);
  printf("\n");
}

int main(void)
{
  unsigned int ux = 0x80000000;
  int x = ux;
  long long unsigned uy = ux;
  long long y = x;

  ux = ux >> 8;
  x  = x >> 8;

  printf("%lu\n", sizeof(ux));
  show_bytes((byte_pointer)&ux, sizeof(ux));
  printf("0x%x 0x%x\n", ux, x);
  printf("%d %lld\n", x>>19, y>>19);

  return 0;
}
```

(a) (2 points) 4

(b) (2 points) 00 00 08 00

(c) (2 points) 00080000 00080000

(d) (2 points) 1 256

First Name: Jae Hong Last Name: Lee BU ID: U27565203

16. (40 points) Assume that the code below is compiled for and executed on a 64-bit little endian computer. Please provide the hex byte value and name of the variable that each address indicated corresponds to. **For addresses that correspond to arrays please indicate the array name and index the address belongs to, e.g.** `str[4]`**. If an address does not correspond to a variable, leave the name blank empty. You can assume that the padded bytes in memory between variables were initialized to 0s. For** `char` **variables the value should be provided as an ASCII character, e.g.** `'a'` **. For all other variables the values should be provided as a two digit hex value.**

```
char  bar[] = "To B or not to";
short sv = 0xBA0D;
int iv  = 0xCAFEF00D;

int * p1 = &iv;
char * p2 = &(bar[3]);

void func()
{
  *p2 = 0x43;
  p2 = (char *)&sv;
  p2 = p2 + 1;
  *p2 = 0x5a;
  *p1 = 0x12345678;
}
```

Variable	Address of Variable
bar	0x0000555555558010
sv	0x0000555555558020
iv	0x0000555555558024
p1	0x0000555555558028
p2	0x0000555555558030

First Name: Jae Hong Last Name: Lee BU ID: U27565203

Given the above fill in the following

(1) 0x0000555555558010: Value: T Name: bar[0]

(2) 0x0000555555558011: Value: 0 Name: bar[1]

(3) 0x0000555555558012: Value: '' Name: bar[2]

(4) 0x0000555555558013: Value: C Name: bar[3]

(5) 0x0000555555558014: Value: '' Name: bar[4]

(6) 0x0000555555558015: Value: 0 Name: bar[5]

(7) 0x0000555555558016: Value: r Name: bar[6]

(8) 0x0000555555558017: Value: '' Name: bar[7]

(9) 0x0000555555558018: Value: 'n' Name: bar[8]

(10) 0x0000555555558019: Value: 'o' Name: bar[9]

(11) 0x000055555555801a: Value: 't' Name: bar[10]

(12) 0x000055555555801b: Value: '' Name: bar[11]

(13) 0x000055555555801c: Value: t Name: bar[12]

(14) 0x000055555555801d: Value: 0 Name: bar[13]

(15) 0x000055555555801e: Value: Name:

(16) 0x000055555555801f: Value: Name:

(17) 0x0000555555558020: Value: 0D Name: SI

(18) 0x0000555555558021: Value: 5e Name: SI

(19) 0x0000555555558022: Value: Name:

First Name: Joe Hong Last Name: Lee BU ID: U25565203

(20) 0x0000555555558023: Value:_____ Name:_____

(21) 0x0000555555558024: Value: 98 Name: iv

(22) 0x0000555555558025: Value: 56 Name: iv

(23) 0x0000555555558026: Value: 34 Name: iv

(24) 0x0000555555558027: Value: 12 Name: iv

(25) 0x0000555555558028: Value: 24 Name: p1 p1

(26) 0x0000555555558029: Value: 80 Name: p1

(27) 0x000055555555802a: Value: 55 Name: p1

(28) 0x000055555555802b: Value: 55 Name: p1

(29) 0x000055555555802c: Value: 55 Name: p1

(30) 0x000055555555802d: Value: 55 Name: p1

(31) 0x000055555555802e: Value: 00 Name: p1

(32) 0x000055555555802f: Value: 00 Name: p1

(33) 0x0000555555558030: Value: 20 Name: p2

(34) 0x0000555555558031: Value: 80 Name: p2

(35) 0x0000555555558032: Value: 55 Name: p2

(36) 0x0000555555558033: Value: 55 Name: p2

(37) 0x0000555555558034: Value: 55 Name: p2

(38) 0x0000555555558035: Value: 55 Name: p2

First Name: Jae Hong Last Name: Lee BU ID: U25505203

(39) `0x0000555555558036`: Value: 00 Name: p2

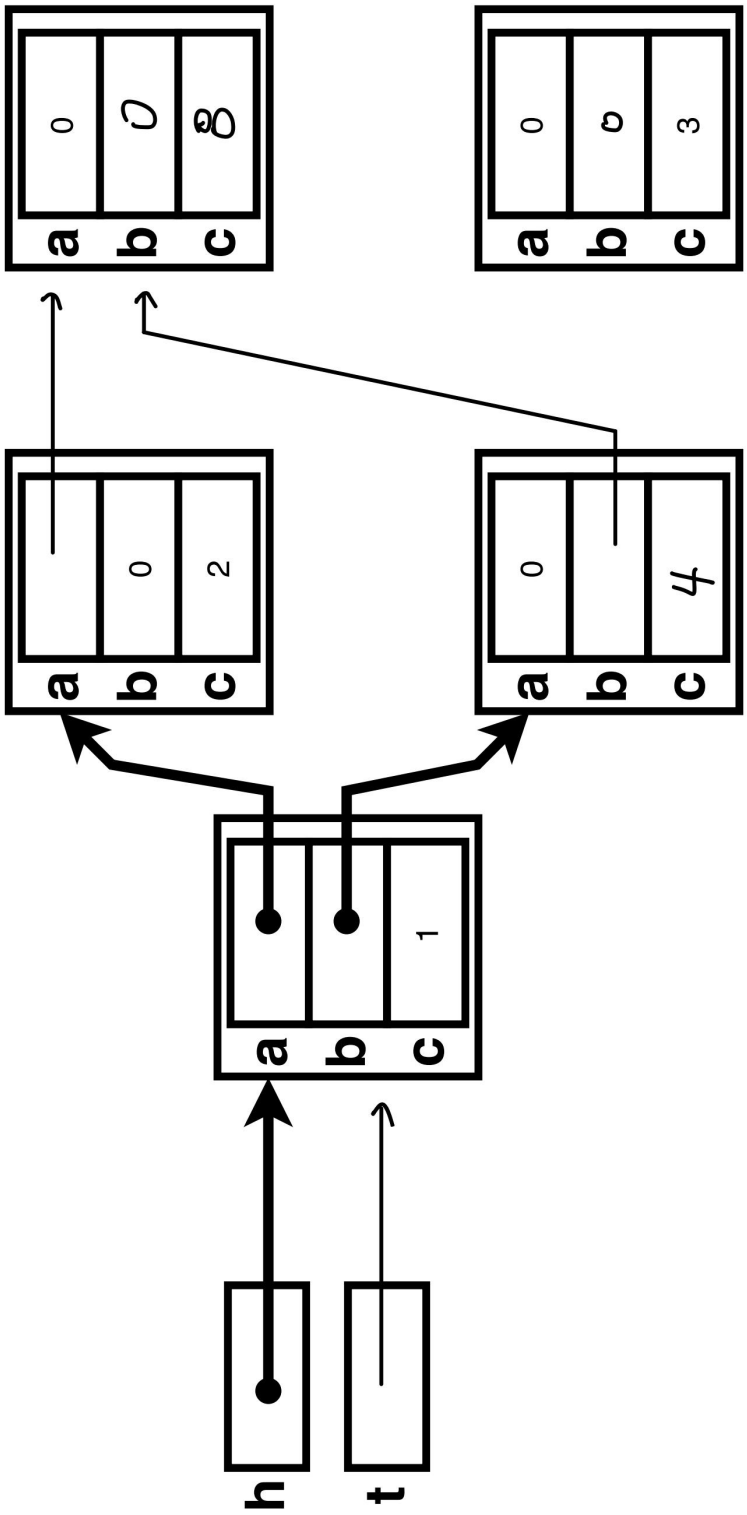
(40) `0x0000555555558037`: Value: 00 Name: p2

First Name: Joe Hong Last Name: Lee BU ID: U20565243

17. (12 points) Given the following code, complete the diagram on the next page assuming main runs to completion. Fill in all blank boxes, either with an arrow or value as needed. A

```
#include <stdlib.h>

struct Node {
  struct Node *a;
  struct Node *b;
  long long    c;
};

struct Node * new()
{
  struct Node *n = malloc(sizeof(struct Node));
  n->a=0; n->b=0; n->c=0;
  return n;
}

struct Node *h;
struct Node *t;

int main(int argc, char **argv)
{
  h = new();
  h->c = 1;
  t = new();
  t->c = 3;
  h->b = new();
  h->b->c = 4;
  t = new();
  h->a = t;
  t->c = 2;
  t = h;
  h->a->a = new();
  t->b->b = h->a->a;
  h->a->a->c = 5;
  t->a->c = 2;
  h->b->b->c = 8;
  return 0;
}
```

First Name: Jae Hong Last Name: Lee BU ID: U27565203



First Name: Jae Hong Last Name: Lee BU ID: U21565203