Systems Programming Mock TCA

Note: Actual TCA has 23 questions not 29.

KU1.2 - Examine specific tools used to solve common build problems

Consider the following 3 files and the output when they are compiled:

users.h

```
#include "toys.h"

typedef struct child_t
{
   int age;
   int height;
   toy own_toy;
} child;
```

toys.h

```
typedef struct toy_t {
   int age;
   double cost;
} toy;
```

main.c

```
#include "toys.h"
#include "users.h"

int main() {
   toy t;
   child brian;
   brian.own_toy = t;
 return 0;
}
```

```
$ gcc -o main main.c
In file included from users.h:1:0,
                 from main.c:2:
toys.h:1:16: error: redefinition of ‘struct toy_t’
 typedef struct toy_t {
                ^
In file included from main.c:1:0:
toys.h:1:16: note: originally defined here
 typedef struct toy_t {
                ^
In file included from users.h:1:0,
                 from main.c:2:
toys.h:4:3: error: conflicting types for ‘toy’
 } toy;
   ^
In file included from main.c:1:0:
toys.h:4:3: note: previous declaration of ‘toy’ was here
 } toy;
   ^
```

Q1. Technically explain the errors in the output and explain why these occur. (2 marks)

Q2. What code you would **add** and **where** would you add it to fix this error?
Note: you cannot change `main.c` or remove anything from the headers. (3 marks)

KU1.5 - Examine typical usage of conditional compilation within a program

Consider the following file (main2.c):

```
#include <stdio.h>
int main(int argc, char argv[]) {
#ifndef TEST
    printf("One");
#else
    printf("Two");
#endif
}
```

Q3. What would be the output if the following two commands are executed? (1 mark)

```
gcc -o main2 –DTEST main2.c
./main2
```

Consider the following code (main2.c):

```
#include <stdio.h>

int main(int argc, char* argv[]) {

    int age;

    printf("Welcome to the Casino\nEnter your age: ");

    scanf("%d", &age);

    if (age < 16)

        printf("You are still too young!");

    else if (age < 25)

        printf("You can only have a drink, no Casino");

    else

        printf("We hope you fully enjoy your stay");
}
```

Q4. Add pre-compiler directives to the code above (by using the blank space between each line or rewriting it on an extra sheet) to produce two versions of the program binary:

i. One that only checks if the age entered is under 16 or not (1 mark)
ii. Another that checks for all three cases. (1 mark)

Q5. Write the GCC commands to compile both versions with different binary names (2 marks)

Command for i

Command for ii

SE1.6 - Evaluate functionality of multisource code using a debug toolchain

Q6. What is a core dump? (1 mark)

Q7. Explain a real-world scenario where core dumps are essential for troubleshooting? (1 mark)

Q8. Given an executable and a core dump, how can you tell what caused the crash? (1 mark)
Note: You are not required to write GDB command, just explain your approach.

Q9. A binary was created with the following command:
`gcc –Wall –DDEBUGVERSION -g -v -no-pie -O0 -o program test.c`
Identify the compiler switches required for debugging with gdb and explain why they are needed. (2 marks)

Consider the following GDB output:

```
GNU gdb (GDB) 7.11.1
...
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 0, 1627400912, 1627400880,
1627401008, 1627400960, 1627401024, 1627400928, 1627400992, 1627400976, 0, 0, 0,
...
Thread 1 "se1" received signal SIGSEGV, Segmentation fault.
0x004011f0 in print_primes (c=65 'A') at coredump.c:10
10                      printf("%d, ", primes[idx++]);

(gdb) list 1,100
1       #include <stdio.h>
2       #include <string.h>
3
4       int idx = 0;
5       int primes[] =
6               {2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47};
7
8       void print_primes(char c) {
9               if (c =='A') {
10                      printf("%d, ", primes[idx++]);
11                      print_primes('A');
12          }
13          else {
14              printf("%d, %d, %d",
15                      primes[0], primes[1], primes[2]);
16          }
17      }
18
19      int main(int argc, char** argv) {
20          if (argc == 1)
21              printf("usage: all | three\n");
22          else if (strcmp(argv[1], "all") == 0)
23              print_primes('A');
24          else
25              print_primes('3');
26      }

(gdb) print idx
$2 = 20473
```

Q10. Which line number did the program crash on? (1 mark)

Q11. Did the user run the program with 'all' or 'three' as a command line argument? Explain your answer. (1 mark)

Q12. What was the value of variable `idx` when the program crashed? (1 mark)

Q13. Explain why the program crashed and suggest a fix. (2 marks)

KU2.1 - Examine typical language scope

Consider the following program (s1.c):

```
#include <stdio.h>
int n1;
int main() {
    int n2;
    printf("Sum=%d", n1+n2);
}
```

Q14. Explain why running this code on different computers may result in different outputs. (1 mark)

Consider the following program (s2.c):

```
1.  float f1 = 99.98;
2.  int function1(float p1) {
3.      p1 *= 99;
4.      return (int)p1;
5.  }
6.  int main() {
7.      float f2 = 789.654;
8.      return function1(f2);
9.  }
```

Q15. Identify the variables in this program and for each specify the line numbers where they are in scope. (1 mark)

Consider the following program (s3.c):

```
#include <stdio.h>

void f(int num) {
    static int test = 0;
    printf("%d.%d\n", num, ++test);
}
int main() {
    f(5);
    f(6);
}
```

Q16. What is the output of this program? Explain your answer (1 mark)

Consider the following program (s4.c):

```
#include <stdio.h>

int inc(int num) {
    return num++;
}
int main() {
    printf("%d", inc(99));
}
```

Q17. What is the output of this program? Explain your answer (1 mark)

Q18. Mention one advantage of using local variables instead of deciding to use only global variables. (1 mark)

KU2.3 - Outline specific use-cases which could require the use of specific memory management

Q19. Unions and structures are similar. Mention something thing they have in common and one difference (1 mark)

Consider the following file (header.h).

```
struct persons {
   int ages[50];
   int lengths[50];
};

union boxes {
   int width[75];
   int length[75];
};
```

Q20. Which of the above date structures use more memory? Explain your answer. (1 mark)

Consider the following lines of code:

```
if (strcmp(s1.xxx, s2->yyy) == 0)
    printf("Match!\n");
```

Q21. What can you tell about s1 and s2 from these lines? (1 mark)

Consider the following code (u1.c).

```
#include <stdio.h>
#include <string.h>
union rectangle {
   int width;
   int height;
};

void main() {
    union rectangle r;
    r.width = 5; r.height = 10;
    printf("Area = %d\n", r.width*r.height);
}
```

Q22. What is the output? (1 mark)

Q23. Explain the output. (1 mark)

SE2.4 - Evaluate the type of program memory management required for a given use-case (structured static or dynamically allocated)

Q24. For the following cases choose if to place your variable on the stack or on the heap? Explain why for each case. (2 marks)

i. A buffer to hold all the contents of a 10MB file
ii. Integer counter of a loop

i

ii

Q25. What are memory leaks? Why should you avoid them? (2 marks)

Consider the following code (memory.c):

```
#include <stdio.h>
#include <stdlib.h>

char* buff;

void foo(char* name) {
    buff = (char*)malloc(256);
    printf("Enter an address: ");
    scanf("%s", buff);
    printf("Enter a name: ");
    scanf("%s", name);
}

int main() {
    char name[128] = {0};
    foo(name);
    return 0;
}
```

Q26. `buff` and `name` are two pointers. For each specify if they contain addresses on the heap or on the stack. (1 mark)

`buff`
`name`

Q27. Identify the memory leak in this code and explain what you would do to avoid it. (2 marks)

Consider the following code (memory2.c):

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.  char* get_text_from_user(int num) {
5.      char *buff = malloc(num);
6.      printf("Enter some text: ");
7.      gets(buff);
8.      return buff;
9.  }
10.
11. int main() {
12.     char* buff;
13.     get_text_from_user(64);
14.     free(buff);
15.     return 0;
16. }
```

Q28: Identify two line numbers where memory is used incorrectly. Justify your answer. (2 marks)

i

ii

Q29. Suggest code changes that would fix these problems. (1 mark)

***