Question 1 — 2%

It's 4pm on Friday. Assignment 1 is due in 7 hours and 59 minutes. A friend invites you to a pub. You reason that one beer would help you focus, so you submit a partial solution and go. Next thing, you wake up with a raging hangover at 2pm Saturday and are in no shape for looking at Secure Programming. On Sunday morning you get back to the assignment, complete and submit it.

Assuming no further communication from you, how will we calculate your mark?

1. Only your partial Friday submission will be marked.
2. Your Sunday submission will be marked with no penalty.
3. Your Sunday submission will be marked, but being more than 24 hours late, the mark is capped at 50%.
4. Your mark is the higher of the partial Friday submission and the capped Sunday submission.

Question 2 — 2%

To reduce potential penalty in the scenario of Question 1, you send the course coordinator an elaborate explanation, involving a relative getting married and your cat defecating on your computer. What is the most likely response?

1. According to University's policies extensions are not granted for weddings. 2. Cats do not defecate on computers.
3. You should grow up.
4. All of the above.

Question 3 — 1%

For the assignment to be marked, the answers should be submitted as:

1. A Microsoft Word document.
2. A typed-up PDF document.
3. A handwritten document scanned to PDF.

4. Any of the above is OK.

Q4. What does the macro-O () do?

A. The macro-O () here pointer, type is expecting the some struct from that struct it will get the value of field and call the address of that value using &. Lastly, it converts all back to the size_t. Here 0 in the O () indicates that from which number values will start getting stores.

Q5. As both structs have the identical fields, one would expect their sizes to be identical. Yet, as you can see, the sizes are different. Please explain why. You may want to use the macro-O () for investigating this.

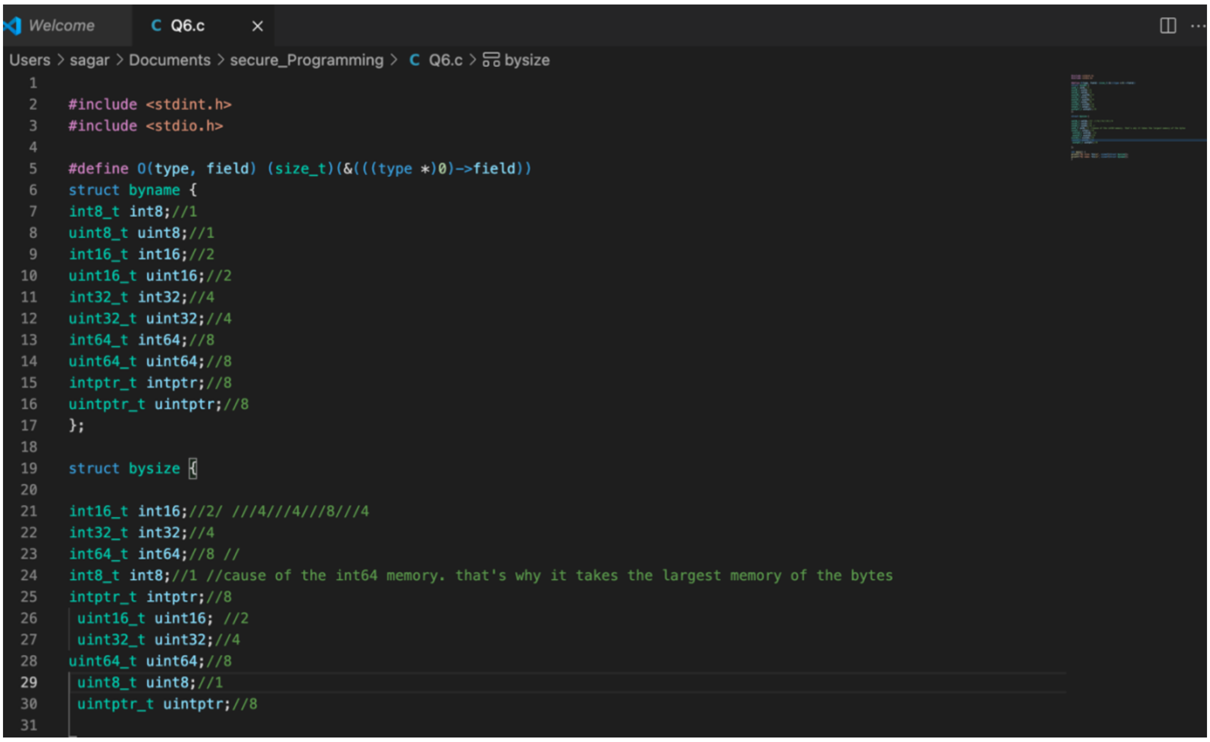
Here, this is the concept of the structure padding. both identical fields show the different fields reason for this different size is using the different format of allocating memory in the memory stack. Architecture of the processor is always reading 4 bytes at a time if the processor is 32 bits.

When we use the bigger value package for storing the value that will stood up to that length of the bytes. When we put the lower value as compared to the package size it will leave the empty space for that package where it cannot be get used.

To describe more,

When we first allocate int64_t it will allocate 8 bytes.

Then when it comes to the int8_t it still allocates 8 bytes even though int8_t's size is 1 byte.


Welcome
Q6.c
Users > sagar > Documents > secure_Programming > Q6.c > bysize

```c

#include <stdint.h>
#include <stdio.h>

#define O(type, field) (size_t)(&(((type *)0)->field))
struct byname {
int8_t int8;//1
uint8_t uint8;//1
int16_t int16;//2
uint16_t uint16;//2
int32_t int32;//4
uint32_t uint32;//4
int64_t int64;//8
uint64_t uint64;//8
intptr_t intptr;//8
uintptr_t uintptr;//8
};

struct bysize {

int16_t int16;//2/ ///4///4///8///4
int32_t int32;//4
int64_t int64;//8 //
int8_t int8;//1 //cause of the int64 memory. that's why it takes the largest memory of the bytes
intptr_t intptr;//8
 uint16_t uint16; //2
 uint32_t uint32;//4
uint64_t uint64;//8
 uint8_t uint8;//1
 uintptr_t uintptr;//8

```

Q7. How much space is wasted in each instance of bysize?

A. There is 2 bytes memory is getting wasted in the mamory stack. Which is happening because of the uint_16 which has size 2 and the next