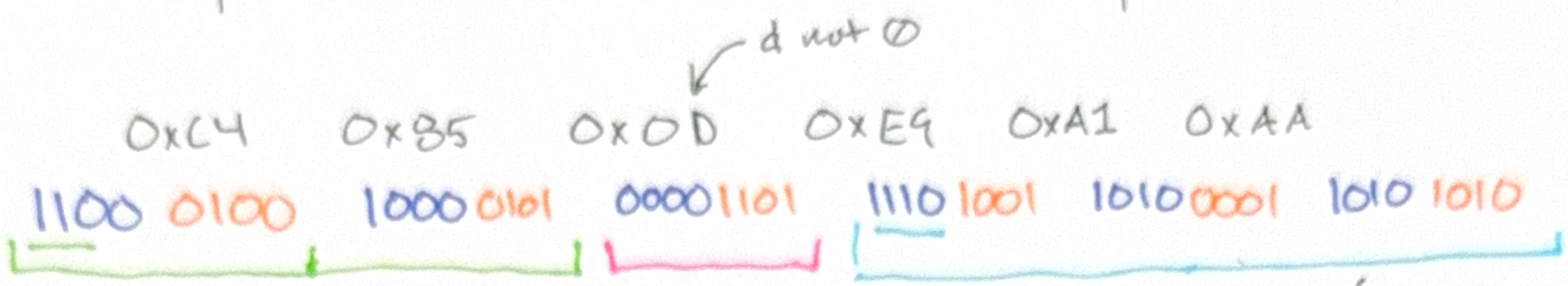
Q1: How many UTF-8 code points are encoded by



Q2: Explain in one sentence what uppercase_ascii does (on handout)

Takes a C string and changes it so lowercase ASCII chars are uppercased.

Q3: Explain in one sentence what truncate_to_n does (on handout)

Takes a C string and a length n and changes the string to be terminated after n characters.

```c
#include <string.h>
#include <stdio.h>
#include <stdint.h>



void uppercase_ascii(char s[]) {
  for(int i = 0; s[i] != '\0'; i += 1) {
    if(s[i] >= 'a' && s[i] <= 'z') { s[i] -= 32; }
  }
}



void truncate_to_n(char s[], uint32_t n) {
  s[n] = '\0';
}

int main() {
  char input[100];
  fgets(input, 100, stdin);
  printf("The input was: %s\n", input);
  truncate_to_n(input, 6);
  printf("First 6 bytes of input: %s\n", input);
  uppercase_ascii(input);
  printf("Input with ASCII uppercased: %s\n", input);
}
```

Problem Statement:
Write a program that takes an input string and prints the first 6 bytes of that string and that string with all of its lowercase ASCII characters uppercased.

What should happen according to the problem statement?

```
$ gcc analyzer.c -o analyzer
$ ./analyzer
utf8 is 🐻
The input was: utf8 is 🐻

First 6 bytes of input: utf8 i

Uppercased: UTF8 IS 🐻
```

What does happen according to this implementation?

```
$ gcc analyzer.c -o analyzer
$ ./analyzer
utf8 is 🐻
The input was: utf8 is 🐻

First 6 bytes of input: utf8 i

Uppercased: UTF8 I
```