Write a function that capitalizes all words of a string.

Prototype: char *cap_string(char *);
Separators of words: space, tabulation, new line, ,, ;, ., !, ?, ", (,), {, and }

```
#include "main.h"
/**
* *cap_string - a function that capitalizes all words of a string.
* @a: char of array
* Return: Always char
* by ramzy
*/
char *cap_string(char *a)
{
int i;
int x;

char arr[] = " \t\n,;.!?\"(){}";

for (i = 0; a[i] != '\0'; i++)
{

if (a[i] >= 'a' && a[i] <= 'z')
{
if (a[i] == a[0])
{
a[0] -= 32;
}
else
{
for (x = 0; arr[x] != '\0'; x++)
{
if (a[i - 1] == arr[x])
{
a[i] -= 32;
}
}
}
}
}

return (a);
}
int main(void)
{
```

```
char str[] = "Expect the best. Prepare for the worst. Capitalize on what
comes.\nhello world! hello-world 0123456hello world\thello world.hello
world\n";
char *ptr;

ptr = cap_string(str);
printf("%s", ptr);
printf("%s", str);
return (0);
}
int main(void)
{
char str[] = "Expect the best. Prepare for the worst. Capitalize on what
comes.\nhello world! hello-world 0123456hello world\thello world.hello
world\n";
char *ptr;

ptr = cap_string(str);
printf("%s", ptr);
printf("%s", str);
return (0);
}
int main(void)
{
char str[] = "Expect the best. Prepare for the worst. Capitalize on what
comes.\nhello world! hello-world 0123456hello world\thello world.hello
world\n";
char *ptr;

ptr = cap_string(str);
printf("%s", ptr);
printf("%s", str);
return (0);
}
int main(void)
{
char str[] = "Expect the best. Prepare for the worst. Capitalize on what
comes.\nhello world! hello-world 0123456hello world\thello world.hello
world\n";
char *ptr;

ptr = cap_string(str);
printf("%s", ptr);
printf("%s", str);
return (0);
}
jint main(void)
{
char str[] = "Expect the best. Prepare for the worst. Capitalize on what
```

comes.\nhello world! hello-world 0123456hello world\thello world.hello world\n";
char *ptr;

ptr = cap_string(str);
printf("%s", ptr);
printf("%s", str);
return (0);
}
int main(void)
{
char str[] = "Expect the best. Prepare for the worst. Capitalize on what comes.\nhello world! hello-world 0123456hello world\thello world.hello world\n";
char *ptr;

ptr = cap_string(str);
printf("%s", ptr);
printf("%s", str);
return (0);
}int main(void)
{
char str[] = "Expect the best. Prepare for the worst. Capitalize on what comes.\nhello world! hello-world 0123456hello world\thello world.hello world\n";
char *ptr;

ptr = cap_string(str);
printf("%s", ptr);
printf("%s", str);
return (0);
}

explain this code as require of

Sure, let's break down the provided code and explain its components.

Function Prototype:

```
char *cap_string(char *);
```

The prototype indicates a function `cap_string` that takes a pointer to a char (essentially a string) and returns a pointer to a char.

Special Characters Array:

```
char arr[] = " \t\n,;.!?\"(){}";
```

This array contains the characters that are considered as word separators. Any character present in this array will signal the function to capitalize the next character if it's an alphabetical lowercase character.

Function `cap_string`:

```
char *cap_string(char *a)
```

The function is designed to capitalize every word in the provided string `a`.

Code Explanation:

1. Loop through each character of the input string `a`:

```
for (i = 0; a[i] != '\0'; i++)
```

2. Check if the current character is lowercase:

```
if (a[i] >= 'a' && a[i] <= 'z')
```

3. If the current character is the first character of the string, capitalize it:

```
if (a[i] == a[0])
{
    a[0] -= 32;
}
```

   The number 32 is the difference between the ASCII value of uppercase and lowercase alphabets.

4. If the current character isn't the first character, loop through the separators array and check if the previous character is a separator:

```
for (x = 0; arr[x] != '\0'; x++)
{
    if (a[i - 1] == arr[x])
    {
        a[i] -= 32;
    }
}
```

   If the previous character is a separator, capitalize the current character.

Finally, the modified string `a` is returned.

Main function(s):

The main function is provided multiple times, possibly by mistake. Let's consider one of them:

```
int main(void)
{
    char str[] = "Expect the best. Prepare for the worst. Capitalize on what comes
    char *ptr;

    ptr = cap_string(str);
    printf("%s", ptr);
    printf("%s", str);
```

```
    return (0);
}
```

1. A string `str` is initialized with a sample text.
2. A pointer `ptr` is initialized.
3. The function `cap_string` is called with `str` as its argument, and its result is stored in `ptr`.
4. The modified string is printed twice using both `ptr` and `str`. Since both are pointing to the same location in memory, the output will be identical.
5. The program returns 0, signaling successful termination.

Errors/Observations:

1. Multiple main functions: There are multiple instances of the `main` function. A valid C program should have only one main function.
2. Typo in `jint main(void)`: The return type `jint` seems to be a typo. It should be `int`.

Overall, the code is designed to capitalize each word in the provided string, where a word is defined as a sequence of characters separated by the special characters in the `arr` array.