SFT - WORKSHOP5

NAME: Chaerin Yoo

ID: 102998234

Email: [cyoo10@myseneca.ca](mailto:cyoo10@myseneca.ca)

Section: NCC

Stringhelp.c

|  |  |
| --- | --- |
| Original Code that has a bug | Fixed Code |
| Line 18: int isNumber(const char\* str, const int len)  {  int i, result = 1;  for (i = 0; i < len && result; i++)  {  result = result && isdigit(str[i]);  }  return result;  } | int isNumber(const char\* str, const int len)  {  int i, result = 1;  for (i = 0; i < len && result; i++)  {  if (!isdigit(str[i]))  {  result = 0;  }  }  return result;  } |

**What was wrong with the lines**: The expression result = result && isdigit(str[i]) can lead to logical errors because the && operator might short-circuit the evaluation. This could cause the loop to exit if result becomes 0, even if there are more characters to check.

**How I fixed it?** Set breakpoints at the start of the isNumber function and stepped through the code line by line. This allowed observation of the behavior of the result variable and how the loop condition was being evaluated.

|  |  |
| --- | --- |
| Original Code that has a bug | Fixed Code |
| while (str[i] != '\0' && isspace(str[i]))  {  if (str[i] == '\n')  {  result.lineStarts[result.numLines] = i + 1;  }  i++;  } | while (str[i] != '\0' && isspace(str[i]))  {  if (str[i] == '\n')  {  result.lineStarts[result.numLines++] = i + 1;  }  i++;  } |

**What was wrong with the lines**: The line result.lineStarts[result.numLines] = i + 1; correctly identifies the start of a new line but does not increment result.numLines.

**How I fixed it?** I tried running the code step by step using a debugger and checked for errors.

|  |  |
| --- | --- |
| Original Code that has a bug | Fixed Code |
| if (isNumber(str + i, sp - i + 1))  {  result.numberStarts[result.numNumbers++] = i;  }  else  {  result.wordStarts[result.numWords++] = i;  } | if (sp > 0 && isNumber(str + i, sp))  {  result.numberStarts[result.numNumbers++] = i;  }  else  {  result.wordStarts[result.numWords++] = i;  } |

**What was wrong with the lines**: The original code calculates the length for isNumber as sp - i + 1, which is incorrect. The correct length is simply sp, because sp already represents the number of characters to the next whitespace or end of string.

**How I fixed it?** I tried running the code step by step using a debugger and checked for errors.

|  |  |
| --- | --- |
| Original Code that has a bug | Fixed Code |
| if (wordNum < idx->numWords && wordNum >= 0)  {  start = idx->wordStarts[wordNum];  sp = nextWhite(idx->str + start);  for (i = 0; i < sp; i++)  {  word[i] = idx->str[i];  }  word[sp] = '\0';  ((struct StringIndex\*)idx)->numWords--;  } | if (wordNum < idx->numWords && wordNum >= 0)  {  start = idx->wordStarts[wordNum];  sp = nextWhite(idx->str + start);  for (i = 0; i < sp; i++)  {  word[i] = idx->str[i];  }  word[sp] = '\0';  //((struct StringIndex\*)idx)->numWords--;  } |

**What was wrong with the lines**: The original code incorrectly decremented numWords in the StringIndex structure, which should not happen when just retrieving a word.

**How I fixed it?** I checked and fixed the error through static analysis in the visual studio.

Main.c

|  |  |
| --- | --- |
| Original Code that has a bug | Fixed Code |
| char testStr[] = { "This is a\n string with embedded newline character and \n12345 numbers inside it as well 67890" }; | char testStr[] = { "This is a\nstring with embedded newline character and \n12345 numbers inside it as well 67890" }; |

**What was wrong with the lines**: Since I did the first line change and there is a space in the result, from the second sentence, there is a space in front of it. So I removed the space to make up for it.

**How I fixed it?** I fixed the result of the awkward part by looking at the debugged compile result and checking the code again.