# EECS 168 2020 Spring Exam 2

Due Date: Monday April 13th 11:59pm

Submit via email to your Lab TA

Place an X in the box that applies to you:

|  |  |
| --- | --- |
| I am in the MWF section (officially) |  |
| I am in the TR section (officially) |  |
| I am an EECS 169 student (leave blank if not in 169) |  |

# Rules (Please Read These!)

* DO NOT alter the formatting of the test. Any changes to formatting could result in grading errors
* Only mark your answers within designated answer boxes
* Read and sign below
* Put your name and KUID on every page

Hi EECS 168 or 169 students. I know you just got this take home exam off the internet, but can you promise me, or more importantly, promise yourself that you'll take this exam without any unauthorized aid?

* Unauthorized aid: google searches, the materials or help of other students, past exams, help from the undergrad staff or GTAs, **any compilers** (e.g. don't just put the code tracing problems in a compiler and run them)
* For the code tracing problem, you are to only read them. They are valid algorithms, though they might not be sensible ones.
* Any strange characters you see are there on purpose.

But since we're in such a crazy world and our class didn't go as planned, I am fine with you using the following authorized aid:

* Authorized resources: your notes, your labs (the code, not the compiler), the online lectures I made for you, your amazing brain

I'd like this exam to still be an assessment of your skill and understanding.

If you agree to this, then please type your name in the box below. If you can't agree to it, then I'm afraid I won't grade your exam. Afterall, I'm not here to test how well you can google things. That's what your job will do.

|  |  |
| --- | --- |
| Type your name |  |

# [20pts] True/False

Place an X in the column of the answer you wish to choose. For example, if you think the answer is true, put an X in the true column.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Statement | True | False |
| 1 | All functions must have parameters |  |  |
| 2 | All functions must return a numeric value |  |  |
| 3 | int main() is a function |  |  |
| 4 | Each parameter a function has requires a type |  |  |
| 5 | Passing an array to a function makes a copy of the entire array |  |  |
| 6 | Passing an int (by value) to a function makes a copy of that int |  |  |
| 7 | The first index in an array is 1 |  |  |
| 8 | The last index in an array is -1 |  |  |
| 9 | Arrays can contain multiple types of data |  |  |
| 10 | A heap allocated 2D array is one contiguous (solid) block of memory with 2 dimensions. Meaning, there are no disjoint pieces. |  |  |

# 

# [20pts] Memory Allocation

Given the code below, indicate where each variable, pointer, array, etc. is allocated, either on the **call stack** or the **heap**.

|  |
| --- |
| int main()  {  int size = 20;  int nums[5];  int\* ptr = new int[size];  int\*\* grid = new int\*[size];  for(int i=0; i<size; i++)  {  grid[i] = new int[ (size/2) ];  }  double temp = 0.0;  char myHeapAllocatedVariable = ‘\0‘;  //main continues... |

|  |  |
| --- | --- |
| Variable, pointer, etc. | Call stack or heap? |
| size |  |
| nums |  |
| nums[2] |  |
| grid |  |
| grid[3] |  |
| grid[2][5] |  |
| temp |  |
| myHeapAllocatedVariable |  |
| ptr |  |
| ptr[17] |  |

# [10pts] Memory management

Finish the definition for the main function such that there are no memory leaks.

int main()

{

int size = 20;

int nums[5];

int\* ptr = new int[size];

int\*\* grid = new int\*[size];

for(int i=0; i<size; i++)

{

grid[i] = new int[ (size/2) ];

}

double temp = 0.0;

char myHeapAllocatedVariable = ‘?‘;

//type your code in the box below

|  |
| --- |
|  |

return(0)

}

# [23pts] Code Tracing

|  |  |
| --- | --- |
| [4pts] Code that runs | Output |
| void func( int& a, int& b ) {  int t = a;  a = b;  b = t;  }  int main() {  int x = 10;  int y = 20;  int t = 5;  func( x , x );  std::cout << x << ‘\n‘;  func( x , y );  std::cout << y << ‘\n‘;  func( y , x );  std::cout << y << ‘\n‘;  func( x , y );  std::cout << t << ‘\n‘;  } |  |

|  |  |  |
| --- | --- | --- |
| nums.txt | [5pts] Code that runs | Output |
| 0  100  200  300  400  500 | //reminder, this is legal  //and written as intended  void func( int& n, int m )  {  n = m;  }  int main()  {  std::ifstream myInFile;  myInFile.open("nums.txt");  int num = 0;  for( int i = 0; i < 500; i=i+0)  {  myInFile >> num;  func( i , num);  std::cout << i << ‘\n‘;  }  myInFile.close();  return( 0 );  } |  |

|  |  |
| --- | --- |
| [4pts] Code that runs | Output to terminal |
| std::string str = "nolemretaw";  int z = str.length()-1;  while( z > -1 )  {  std::cout << str.at(z);  z--;  }  if( str == "watermelon" )  {  std::cout << "yum";  } |  |

|  |  |
| --- | --- |
| [2pts] Code that runs | Output to terminal |
| //reminder, this is legal  //and written as intended  int doAllTheWork()  {  int x = 99;  int y = -1;  int z = 0;  return(5);  std::cout << x << ‘\n‘;  std::cout << y << ‘\n‘;  std::cout << z << ‘\n‘;  x = y + 10;  y = x\*x;  z = y + x + x +5;  z = z + 1;  std::cout << x << ‘\n‘;  std::cout << y << ‘\n‘;  std::cout << z << ‘\n‘;  return(z);  }  int main()  {  int x = doAllTheWork();  std::cout << x << "\n";  std::cout << "Goodbye.\n";  return(0);  } |  |

|  |  |
| --- | --- |
| [4pts] Code that runs | Output to terminal |
| void matrix(int n)  {  n = -1;  std::cout << "Hahahah!\n";  }  int main()  {  int x = 99;    std::cout << x << ‘\n‘;  matrix(x);  std::cout << x << ‘\n‘;  } |  |

|  |  |
| --- | --- |
| [4pts] Code that runs | Output to terminal |
| int size = 4;  int\* nums = new int[size];  int\* nums2 = new int[size];  for(int i=0; i<size; i++)  {  nums[i] = i;  nums2[i] = (10\*i)+1;  }  int\* temp = nums;  temp[size-1] = nums2[0];  for(int i=0; i<size; i++)  {  std::cout << nums[i] << ‘\n‘;  } |  |

# [27pts] Code Writing

[12pts] Write only the definition for a function named *fizzBuzz*. It takes an integer, *n*, as a parameter and returns a std::string. If the number is a multiple of 3, return the string “fizz”. If the number is a multiple of 5, return the string “buzz”. If the number is a multiple of both 3 and 5, return the string “fizzbuzz”. Otherwise return the string “boring”.

Write your code in the designated box on the following page

|  |
| --- |
| //Function definition below |

[15pts] Write just a function definition for a function called *isLeapYear*. It takes an int representing a year as a parameter and returns true if the year is a leap year.

Rules for being a leap year:

* All years that are multiples of 400 are leap years (e.g. 2000)
* All years that are multiples of 100 are **not** leap years (e.g. 1300) unless they are also multiples of 400
* All other years that are multiples of 4 are leap years (e.g. 2016)
* All other years are not leap years

Write your code in the designated box on the following page

|  |
| --- |
| //function definition below |

PAGE 14 SHOULD BE THE FINAL PAGE OF THE EXAM. CHECK THE PAGE NUMBER BELOW. IF IT'S NOT PAGE 14, PLEASE CORRECT ANY FORMATTING ISSUES

Thank you so much for taking my test.