

NAME: \_\_\_\_\_

BU ID: \_\_\_\_\_

### Sample Midterm Exam


EC327 Introduction to Software Engineering

---

**Total: 100 points + extra credit (5 points)**

**\*\*MAKE SURE TO READ THE FOLLOWING BEFORE STARTING YOUR EXAM\*\***

#### Exam Rules:

- **Save your work often during the exam.** You would not want to lose any work.
- You **CAN** use the following during the exam:
  - o Your own 1-page cheat-sheet (can be double-sided).
    - No sharing!
  - o Linux programs (gcc, vi, gedit, emacs, and so on).
  - o Course material (lecture notes, solutions, etc.) posted at: [learn.bu.edu](http://learn.bu.edu)  EC327 Spring2020
  - o [www.cplusplus.com](http://www.cplusplus.com)
- You **CANNOT** use any personal electronic devices (including hardware calculators, cell phones, laptops, tablets, or others) or any other books or printouts.
- You have until **4:30pm (SHARP)** to finish and submit your exam. Please follow the submission instructions below. The submission site will automatically close after the deadline, so be sure to submit on time.
- Do not spend too much time on any one question, you can always return to it later.
- You **MUST** use g++ and compile with **-std=c++11** otherwise your code may not get any points!

**GOOD LUCK!**

***READ BEFORE YOU CONTINUE:***

- You are allowed to submit your exam zip file multiple times. **Only your latest submission will be graded.**
- Make sure to follow the folder naming convention outlined above (e.g., *jbond\_007\_midterm.zip*).
- Make sure to put all the code and all the files you need to submit in your exam directory, e.g., in *jbond\_007\_midterm*. Download and check what you submitted to make sure your files are there.
- Note that you can also access your midterm exam directory on the Linux machine by typing its path in a browser. You can then drag-and-drop files into it.
- **You are not allowed to access any other directories on the engineering grid (such as /ad/eng folders) or ssh to any machines.**
- **Similarly, you must not visit any other course website on Blackboard and strictly use it to view EC327 content.**
- All actions on the lab machines are being logged during the exam. Students not following exam rules may be asked to leave and not receive a grade for the exam.
- **Save your work often!**

## Q1. True / False and Multiple Choice [15 points, 5 each]

---

**Write your answers on the exam paper.**

### 1.a.

Determine whether the following statement is **true or false**. If you select false, briefly state **why**.

Given the code snippet:

```
#ifdef VERBOSE
cout << "Performing iteration number " << j << endl;
#endif
```

The cout statement is compiled in the code only if VERBOSE was defined earlier in the program.

### 1.b.

Consider the following code:

```
void foo(int x, int &y, char* z) {
    int array = new int[5];
    static int st=1;
    char c = 'A';
    for (int i=0; i<x; i++) {
        st += y;
    }
}
```

Which of the following would definitely be stored on the stack? Choose one or more of the multiple choices.

- i. x
- ii. y
- iii. z
- iv. array
- v. st
- vi. c

**1.c.**

Determine whether the following statement is **true or false**. If you say false, briefly state **why**.

The following C++ program

```
int main() {  
    for (int j=0; j < 5; j++) {  
        cout << j << endl;  
    }  
    return 0;  
}
```

Will print the following output in the terminal:

```
1  
2  
3  
4  
5
```

## Q2. Short Answers [15 points, 5 each]

---

*Write your answers on the paper exam.*

### 2.a.

Consider the following Simplified Intel Assembly code. What value would be stored in memory location 20 after the execution of the code is complete?

*This code uses the Simplified Intel Assembly we covered in class (available online during the exam).*

```
    mov R0, 7
    mov R1, 5
    mov R2, 0
    mov R3, 0
loop: cmp R1, R2
      je end
      inc R2
      add R3, R0
      jmp loop
end:  inc R3
      mov R4, 20
      mov [R4], R3
      halt
```

### 2.b.

Convert the following decimal number into hexadecimal and the hexadecimal number into decimal. Show your work when providing the solution (i.e., the steps you followed to get to the results).

a. 0x0F2A (convert to decimal)

ii. 287 (convert to hexadecimal)

### 2.c.

Consider the following code snippet running on a 32-bit machine:

```
int value = 5;
int* p = &value;
cout<< p <<endl; //prints out 0x7fff560b
```

For the following statements, provide the values of value, \*p, and p after executing the statement. Before each of the statements execute, assume the following initialization is repeated.

```
value= 5;
p = &value;
```

	value	*p	p
i. value += 2;			
ii. *p++;			
iii. ++p;			
iv. *p = value + p;			

\* You can write “unknown” if the value at a particular address or an address is not known.

### Q3. Functions, Arrays, Loops [15 points]

---

**Files to submit:** Q3.cpp

Write C++ code implementing the following function.

Submit only the cpp file for the function. Do NOT submit the main function or the header file. You can find the header file on Blackboard (Q3.h), along with a sample main (Q3main.cpp).

*\*Your code (when compiled with appropriate test code and the necessary standard libraries) needs to compile in order to receive any partial credit.\**

Write a function `printEven` that given an array of integers, prints on the screen the elements of that array that are even, as follows:

```
void printEven(int numbers[ ], int numsize);
// Given an integer array of size numsize, this function prints the
```

```
// elements of the array that are even, one per line
```

For example, if the following array was provided to the function:  
`int array[5] = {0,3,6,75,8};`

Then `printEven(array, 5)` would print this on the terminal:

```
0
6
8
```

#### Q4. Arrays and strings [20 points]

---

##### **Files to submit: Q4.cpp**

Write C++ code implementing the following function.

Submit only the cpp file for the function. Do NOT submit the main function or the header file. You can find the header file on Blackboard (Q4.h), along with a sample main (Q4main.cpp).

*\*Your code (when compiled with appropriate test code and the necessary standard libraries) needs to compile in order to receive any partial credit.\**

Write a `nthWord` function that, given a c-string as an argument, returns the n-th word in that string as a new string. The prototype of `nthWord` is as follows:

```
char* nthWord(char word[], int n);  
\\returns a pointer to a string containing the n-th word in the string  
word.
```

For example, if called as `nthWord("This is a hard midterm",5)`, the function will return a pointer to a new c-string containing "midterm".

Note that this function should create a new c-string, it is not enough to return a pointer to a location inside the `word` string passed as an argument.

#### Q5. File I/O [20 points]

---

##### **Files to submit: Q5.cpp**

Write C++ code implementing the following functions.

Submit only the cpp file for the function. Do NOT submit the main function or the header one. You can find the header file on Blackboard (Q5.h), along with a sample main (Q5main.cpp).

*\*Your code (when compiled with appropriate test code and the necessary standard libraries) needs to compile in order to receive any partial credit.\**

You are given a file `grades.txt` with the following format:

John Doe EC327 A  
Bruce Wayne EC327 C-  
Dorothy Gale EC330 A+  
Kevin Mitnick EC521 B-  
John Doe EC330 C+

Each line of the file contains the first name and the last name of a student, the class they took, and the grade they received.

Write a function `transcript` that given a student's name creates a file `transcript.txt` containing all the classes that the student took and their respective grades. The prototype of `getGrade` is as follows:

```
int transcript(string first, string last);  
\\Given the first and last name of a student, extracts all the classes  
that the student took and lists them in a new file transcript.txt
```

Note that each execution of the function should create a new `transcript.txt` file. In other word if there was a `transcript.txt` already, the file should be overwritten. The format of the transcript file should be as follows:

EC327 A  
EC330 C+

The function `transcript` should return 0 if the student was found and a transcript was generated, and -1 otherwise. The file `transcript.txt` should **not** be created if the student is not found.

## Q6. Mathematical Functions, recursion [10 points]

### ***Files to submit: Q6.cpp***

Write C++ code implementing the following functions.

Submit only the cpp file for the function. Do NOT submit the main function or the header file. You can find the header file on Blackboard (Q6.h), along with a sample main (Q6main.cpp).

*\*Your code (when compiled with appropriate test code and the necessary standard libraries) needs to compile in order to receive any partial credit.\**

The binomial coefficient  $\text{bin}(n, k)$  of two positive integers  $n$  and  $k$  can be defined recursively as

$$\begin{aligned}\text{bin}(n, 0) &= 1 \\ \text{bin}(n, n) &= 1 \\ \text{bin}(n, k) &= \text{bin}(n-1, k-1) + \text{bin}(n-1, k)\end{aligned}$$



Write a recursive function `bin` that returns the binomial coefficient of two positive integers. The prototype of `bin` is:

```
int bin(int n, int k);  
\\Returns the binomial coefficient of n and k
```

You don't have to consider the case in which the function is given negative numbers.

### Extra credit [5 points]

#### Files to submit: `extra.cpp`

Write C++ code implementing the following functions.

Submit only the cpp file for the function. Do NOT submit the main function or the header file. You can find the header file on Blackboard (`extra.h`), along with a sample main (`extramain.cpp`).

*\*Your code (when compiled with appropriate test code and the necessary standard libraries) needs to compile in order to receive any partial credit.\**

Write a function `toBin` that takes a string as input and returns a string containing the binary representation of the string. For example, for the string "hello", `toBin("hello")` should return a pointer to a string containing "0110100001100101011011000110110001101111". The prototype of `toBin` is as follows:

```
char *toBin(string s);
```

- Turn in your paper exam copy. **Only Q1 and Q2 will be graded on the paper.**
- Your midterm folder on the lab machine should contain:
  - Q3.cpp
  - Q4.cpp
  - Q5.cpp
  - Q6.cpp
  - extra.cpp
- **Zip your exam folder and submit** following the instructions provided.
- **Make sure all of your code compiles!**
- **Before logging off, check on the submission portal that your submission is correct and includes all your files!**