

**UNIVERSITY OF TORONTO  
FACULTY OF APPLIED SCIENCE AND ENGINEERING**

**APS 105 — Computer Fundamentals  
Midterm Examination  
February 27, 2024, 1:10 p.m. – 2:50 p.m. (100 minutes)  
Examiners: S. Emara, J. Eyolfson, B. Korst**

This is a “closed book” examination; no aids are permitted. No calculators or other electronic devices are allowed.

All questions are to be answered on the examination paper. If the space provided for a question is insufficient, you may use the last page to complete your answer. If you use the last page, please direct the marker to that page and indicate clearly on that page which question(s) you are answering there.

You must use the C language to answer programming questions. You are not required write `#include` directives in your solutions.

Note that the summary of the total marks for the questions and the marks given for the questions is on the last page.

The examination has 14 pages, including this one.

**Question 1** [3 Marks]

Write a single C statement that declares a bool called isValidIndex and initializes its value to true if and only if an integer, index, is a valid index within an array of size, arrayLength.

**Solution:** `bool isValidIndex = index >= 0 && index < arrayLength;`

**Question 2 [3 Marks]**

Write a single C statement that declares a bool variable named `myBool` and sets its value to true if and only if the value stored in an integer variable named `myInt` is an even negative number. You may assume that the variable `myInt` has been declared and initialized already to some value.

**Solution:**

```
bool myBool = ( (myInt % 2 == 0) && (myInt < 0));
```

**Question 3 [3 Marks]**

Write a single C statement that declares an int variable called randomNum and sets it to a random number that is a multiple of 5 and is between 5 and 25 (inclusive). This means that randomNum should be set to either: 5, 10, 15, 20 or 25. You may use the rand() function.

**Solution:**

```
int randomNum = ((rand() % 5) + 1) * 5;
```

**Question 4 [3 Marks]**

The following C function takes a single integer argument, grade, that represents a grade between 0 and 100 (inclusive) and prints a letter grade. You decide to test the program by calling `printLetterGrade(75)`, what output would you see as a result?

```
void printLetterGrade(int grade) {  
    if (grade >= 80) {  
        printf("A\n");  
    }  
    if (grade >= 70) {  
        printf("B\n");  
    }  
    if (grade >= 60) {  
        printf("C\n");  
    }  
    if (grade >= 50) {  
        printf("D\n");  
    }  
    else {  
        printf("F\n");  
    }  
}
```

**Solution:**

- B
- C
- D

### Question 5 [5 Marks]

Indicate whether the statements below are **True** or **False** by circling the correct choice.

1. The operator == can be used to assign a value to a variable.  
**True**                      **False**
2. It is valid to dereference an uninitialized pointer.  
**True**                      **False**
3. A boolean variable is typically stored using 1 byte.  
**True**                      **False**
4. If arr is an array of integers and p is a pointer to int variable, then \*p == a[0] assigns the first value of the array to p.  
**True**                      **False**
5. One of the roles of the compiler is to indicate logic errors in C language.  
**True**                      **False**
6. The sizeof(double) expression returns the size in bytes of data type double.  
**True**                      **False**
7. In functions, "pass by value" refers to the process where the function receives a direct link to the original parameter, rather than a copy of its value.  
**True**                      **False**
8. After the declaration of an array, the name of the array itself is a pointer to the first element of the array.  
**True**                      **False**
9. When an array is used as an argument in a function call, a copy of that array is passed to the function.  
**True**                      **False**
10. The function rand() needs a different "seed" every time you run your program to generate truly random numbers.  
**True**                      **False**

### Solution:

Solution

- 1- False
- 2- False
- 3- True
- 4- False
- 5- False
- 6- True
- 7- False
- 8- True
- 9- False
- 10- True

**Question 6 [8 Marks]**

What's the output of the following program?

```
int main(void) {
    int first = 1;
    int arr[4] = {5, 10, 15, 20};
    int* p = arr + 1, *q = arr + 3;
    int* s = &first;
    *s = q - p;

    arr[first] = (*arr) + 2;

    *q = *p + 5;

    (*p)++;

    printf("first = %d\n", first);
    for (int i = 0; i < 4; i++) {
        printf("%d, ", *(arr + i));
    }

    return 0;
}
```

**Solution:**

```
first = 2
5, 11, 7, 15,
```

### Question 7 [7 Marks]

Write a C main function that takes in two dates from the user, and determines if the first date or the second date is later or if they are the same date. Three examples are shown below. User input is in **bold**.

#### Example 1:

Enter first date (day/month): **22/10**

Enter second date (day/month): **1/10**

First date is later

#### Example 2:

Enter first date (day/month): **22/10**

Enter second date (day/month): **22/11**

Second date is later

#### Example 3:

Enter first date (day/month): **22/10**

Enter second date (day/month): **22/10**

It's the same date

#### Solution:

```
#include <stdio.h>

int main(void) {
    int day1, month1, day2, month2;
    char c;
    printf("Enter first date (day/month): ");
    scanf("%d%c%d", &day1, &c, &month1);

    printf("Enter second date (day/month): ");
    scanf("%d%c%d", &day2, &c, &month2);

    if (month2 > month1) {
        printf("Second date is later\n");
    } else if (month1 > month2) {
        printf("First date is later\n");
    } else if (day2 > day1) { // same month
        printf("Second date is later\n");
    } else if (day2 < day1) {
        printf("First date is later\n");
    } else {
        printf("It's the same date\n");
    }
    return 0;
}
```

**Question 8 [8 Marks]**

Write a complete C function called `sumDivisors`, the prototype of which is given below, that accepts one parameter named `num` of type `int`. The function must return the sum of all divisors of `num`. You may assume that the argument passed to the function is a positive integer. For example, if `num` was 6, the divisors of 6 are 1, 2, 3. The function should return  $1 + 2 + 3 = 6$ . The divisors do not include the number itself, but they include 1.

**Solution:**

```
int sumDivisors(int num) {  
    int sum = 0;  
    for (int i = 1; i < num; i++) {  
        if (num % i == 0) {  
            sum += i;  
        }  
    }  
    return sum;  
}
```

**Question 9 [10 Marks]**

Write a C function named `secondLargest` that takes two arguments: an array of positive integers and the size of the array. The function should return the second largest element in the array. Assume that the array always has at least two elements. For example, given the array {3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5} with length 11, the function should return 6. If all elements have the same value, the function should return 0.

**Solution:**

```
/* Solution that doesn't use unique elements */
int secondLargest(int array[], int arrayLength) {
    int largest;
    int nextLargest;
    if (array[0] > array[1]) {
        largest = array[0];
        nextLargest = array[1];
    }
    else {
        largest = array[1];
        nextLargest = array[0];
    }
    for (int i = 2; i < arrayLength; ++i) {
        if (array[i] > largest) {
            nextLargest = largest;
            largest = array[i];
        }
        else if (array[i] > nextLargest) {
            nextLargest = array[i];
        }
    }
    return nextLargest;
}

/* Alternative Solution that checks for unique elements */
int secondLargestAlt(int array[], int arrayLength) {
    int largest = array[0];
    int nextLargest = 0;
    for (int i = 1; i < arrayLength; ++i) {
        if (array[i] > largest) {
            nextLargest = largest;
            largest = array[i];
        }
        else if (array[i] != largest && array[i] > nextLargest) {
            nextLargest = array[i];
        }
    }
}
```

```
    return nextLargest;  
}
```

**Question 10 [12 Marks]**

Write a C main function that takes in the number of lines, and prints the following pattern depending on the number of lines entered by the user.

**Example 1**

Enter number of lines: 5

```
*****
 *  *
  *
 *  *
*****
```

**Example 2**

Enter number of lines: 6

```
*****
 *  *
 **
 **
 *  *
*****
```

**Solution:**

```
void printPattern(int lines) {
    for (int row = 1; row <= lines; row++) {
        for (int col = 1; col <= lines; col++) {
            if (row == 1 || row == lines || col + row - 1 == lines || col == row) {
                printf("*");
            } else {
                printf(" ");
            }
        }
        printf("\n");
    }
}

int main(void) {
    int lines;
    printf("Enter number of lines: ");
    scanf("%d", &lines);
    printPattern(lines);
    return 0;
}
```

**Question 11 [13 Marks]**

Write a complete C function called `arrayInArray`, the prototype of which is given below. The function accepts two arrays of type `int` as parameters, named `a` and `b`. The sizes of `b` and `a` are unknown, but the last element in both array is always `-1`. All other elements in the arrays are positive integers. The function must return `true` if the sequence of numbers in array `b` is found in array `a` excluding the last element in the array, which is always `-1`.

For example, if `a = {6, 8, 7, 6, 1, 7, 4, -1}` and `b = {6, 1, -1}`, the function returns `true` as the sequence of `{6, 1}` is available in `a` at index 3.

Another example, if `a = {6, 8, 7, 6, 1, 7, 4, -1}` and `b = {7, 8, -1}`, the function returns `false` as the sequence of `{7, 8}` is not available in `a`.

**Solution:**

```
bool arrayInArray(int a[], int b[]) {
    bool found = false;
    for (int i = 0; a[i] != -1; i++) {
        if (a[i] == b[0]) {
            found = true;
            for (int j = 0; b[j] != -1 && a[i + j] != -1 && found; j++) {
                printf("i = %d, j = %d\n", i, j);
                if (a[i + j] != b[j]) {
                    found = false;
                }
            }
            if (found) {
                return found;
            }
        }
    }
    return found; // or return false;
}
```

### MARKS

1	2	3	4	5	6	7	8	9	10	11	Total
/3	/3	/3	/3	/5	/8	/7	/8	/10	/12	/13	/75

*The balance of this page has been left blank intentionally. You may use it for answers to any question in this examination.*