Programming II (CCINF1002) 2025/2026

Model Exam Duration: 2h

Student ID: | | | | | | | | | |

Full name: _____

Group 1 (50%) – C Fundamentals

1.1. What will be the result of the following expression? `7 % 3` [ ]

1.2. What will be the result of the following expression? `5 + 6 * 3 / 4` [ ]

1.3. Classify each of the following 4 names: **yes** if it is a valid identifier for a variable name and **no** if it is not.

`break123` [ ] `123break` [ ] `break` [ ] `_123break` [ ]

1.4. Complete the blank space so that the following code reads an integer and puts it on variable `x`:

```
int x;
scanf( [          ] );
```

1.5. Complete the blank space so that the following code snippet prints exactly `1.50`:

```
float f = 1.5;
printf( [          ] );
```

1.6. What is the output of the following code snippet? [ ]

```
int i = 2;
int j = 4;
int k = ++i * j++;
printf("%d %d %d\n", i, j, k);
```

1.7. What is the output of the following code snippet? [ ]

```
int a = 1;
int b = 0;
int c = 1;
if ( (a || b) && c ) printf("true");
else printf("false");
```

1.8. How many times will the following code print `"Hello"`? ☐

```
int n = 1;
while (n < 8) {
   printf("Hello");
   n *= 2;
}
```

1.9. What will be the value of the variable `count` after the following code snippet? ☐

```
int n = 5, count=0;
for (int i=0; i<n; i++)
   for (int j=n; j>=0; j--)
      count++;
```

1.10. What will be the value of the variable `count` after the following code snippet? ☐

```
int n=10, count = 0;
for (int i=0; i<20; i++) {
   if (i>3 && i<7) continue;
   if (i>10) break;
   count++;
}
```

1.11. Assume the `add` function is defined as follows:

```
void add(int x, int k) {
   x += k;
}
```

What is the output of the following code snippet? ☐

```
int x = 1, k = 2;
add(x, k);
printf("%d", x);
```

1.12. What is the value returned by a call to `foo(5)`? ☐

```
int foo(int x) {
   if (x==0) return 0;
   else return x + foo(x-1);
}
```

1.13. What is the value returned by a call to `secret(30, 12)`? ☐

```
int secret(int a, int b) {
   int tmp;
   while (b != 0) {
      tmp = a % b;
      a = b;
      b = tmp;
   }
   return a;
}
```

1.14. What is the output of the following code snippet? ☐

```
int a[] = {2,4,6,8};
printf("%d", a[2]);
```

1.15. What will be the content of array `a[]` after the following code snippet? ☐ ☐ ☐ ☐ ☐

```
int a[] = {5,4,3,2,1};
for (int i=1; i<5; i++) a[i] = a[i-1] + 1;
```

1.16. Complete the blank spaces so that the following code snippet prints the number `4`:

```
int m[3][3] = { {1,2,3}, {4,5,6}, {7,8,9} };
printf("%d", m[ ][ ];
```

1.17. What is the output of the following code snippet? [ ]

```
char c1 = 'a';
char c2 = c1 + 3;
printf("%c", c2);
```

1.18. What is the output of the following code snippet? [ ]

```
char str[] = "hello123";
int i = 0;
while (!isdigit(str[i])) i++;
printf("%d", i);
```

1.19. Complete the blank space so that the following function returns the last character of a string `str`: *(you can use `string.h` functions)*

```
char lastChar(char str[]) {
    return                          ;
}
```

1.20. Complete the blank space so that the following function returns a *true* value if the strings *a* and *b* are equal (that is, have the same content) or a *false* value otherwise: *(you can use `string.h` functions)*

```
int equal(char a[], char b[]) {
    return                          ;
}
```

Group 2 (50%) – Creating Code

2.1. (11%) Write a **complete C program** (with *includes* and the *main* function) that reads two integers *a* and *b* and prints all the even numbers between *a* and *b* (inclusive) separated by single spaces and ended with a newline character.

Example: If the user inputs `2` and `11`, your code should print: `2 4 6 8 10`

2.2. (11%) Write a **function** `int count_digits(int n, int d)` that returns the number of times digit d appears in the integer n. You can be assured that d will always be an integer between 0 and 9.

Example 1: Calling `count_digits(122321, 2)` should return 3 (digit 2 appears three times)

Example 2: Calling `count_digits(4242, 5)` should return 0 (digit 5 does not appear)

2.3. (11%) Write a **function** `void swap(int a[], int n)` that receives an array a with n integers an swaps all pairs of consecutive positions, changing the array itself: the 1st element should be swapped with the 3nd element, the 3rd with the 4th, etc. If the n is odd, the last element should stay in the same position.

Example: Calling it with $a = \{2, 4, 6, 8, 10\}$ and $n = 5$ should change the contents of the array to $\{4, 2, 8, 6, 10\}$.

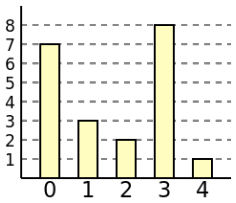
2.4. (11%) Write a **function** `int palindrome(char str[])` that receives a string *str* of letters and returns 1 if it is a palindrome and 0 otherwise. A string is considered a palindrome if it reads the same backwards as forwards. A lowercase letter should be considered the same as its uppercase version. *(you will receive partial points if your code does not take care of mixed case strings but works if all the letters are in lower or upper´ case)*

Example 1: Calling `palindrome("anna")` should return 1

Example 2: Calling `palindrome("Radar")` should return 1

Example 3: Calling `palindrome("WoWo")` should return 0

2.5. (6%) Consider an array h with n distinct integers representing heights of towers. For instance, the array $h = \{7, 3, 2, 8, 1\}$ represents the figure on the right. Let's call $score(i)$ to the distance to the closest tower that is higher than $h[i]$, that is:



$$score(i) = min\{|i - j| : \text{for every } j \text{ such that } h[j] > h[i]\}$$

This value of $score(i)$ for the highest tower is 0. For the figure on the right we have:

- $score(0) = 3$ (the closest highest tower is on position 3 and $|0 - 3| = 3$)
- $score(1) = 1$ (the closest highest tower is on position 0 and $|1 - 0| = 1$)
- $score(2) = 1$ (the closest highest towers are on positions 1 or 3)
- $score(3) = 0$ (this is the highest tower)
- $score(4) = 1$ (the closest highest tower is on position 3 and $|4 - 3| = 1$)

Write a **function** `int total_score(int h[], int n)` that computes the total score of the array, that is, the summation of all scores. More formally, it should return $\sum_{i=0}^{n-1} score(i)$.

To obtain full score your solution must be efficient in terms of computation time.
(able to solve a case with $n = 200\,000$ in less than 1 second in a normal laptop)

Example: Calling it with $h = \{7, 3, 2, 8, 1\}$ and $n = 5$ should return 6
(this is the example explained in the question statement)

(answer to question 2.5 – `total_score` function)

I hope you have enjoyed this course and learned many new things!

GOOD LUCK ON THE EXAM SEASON