

Moch Exam October 2021

Programming 1 exam, Fall 2020

- This exam consists of 5 tasks with a total of 100 points.
- You can write all your code in a .cpp file or use header files (.h).
- Make sure you include all your header files if you use any.
- Make a zip file of your project folder.
- Write your comments in English.
- Using functions make your code modular
- You are free to use you own method to solve the problems
- Double check your zip folder once before submitting.

Task 1: (10 points)

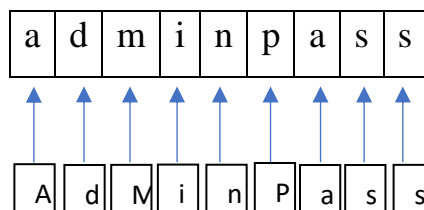
You need to define a const string variable called password and initialize the value “adminpass” to it.

```
password = "adminpass";
```

then when you run your program, you need to ask the user to enter a password.

Your task is to check the strings **character by character** to see if two values are the same.

The password is **not** case sensitive, which means if the user enters AdMinPass or ADMINPASS or another form, the password will still be correct.



Do not compare strings or use standard libraries for this task.

If the values are the same which means password is correct, then go to task 3 and if it's not correct keep asking user to enter the password. If the user enters the password wrong 5 times, the program will be terminated.

You can use `std::exit(0);` or any other methods you know to terminate the program.

Task 2: Initial and final characters (10 points)

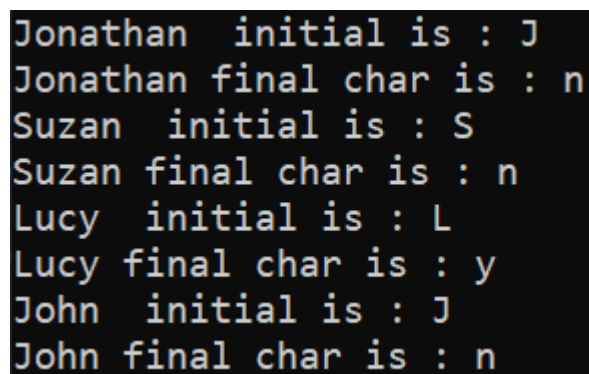
You have a vector of strings that has the following names in it.

```
std::vector<std::string> names{ "Jonathan", "Suzan", "Lucy", "John"};
```

Your task is to print out initial and Last character of each names in the vector.

Following figure illustrates an example of what you should get as a result on console.

Names above are just examples. This task should be dynamic, so whatever user enters in this vector, the output always will print initial and last characters.



```
Jonathan initial is : J
Jonathan final char is : n
Suzan initial is : S
Suzan final char is : n
Lucy initial is : L
Lucy final char is : y
John initial is : J
John final char is : n
```

Task 3. Main Menu (20 points)

The layout and look of your menu are optional but it should at least have the following options:

Main Menu:

1. Print names' Initials and Lasts (Task 2)
2. Students' exam result (Task4)
3. Typing a sentence (Task5)
4. Terminate the program (Quit)

Your menu should look something like the following figures:

```
| 1: task 1
| 2: task 2
->| 3: task 3
| 4: task 4
```

Figure 1: Task 3 has been selected

Arrow moves up and down (using arrow keys or W and S keys) to select which task you want to run

```
->| 1: task 1
| 2: task 2
| 3: task 3
| 4: task 4
```

Figure 2: Task 1 has been selected

Here it shows that the arrow is pointing to task 1, if the user presses enter Task 1 (Function task 1) will be called.

Note: if you fail to do this Task with this method you can do it any other way you can, but you may lose some points not doing the task this way.

```
1: task 1
2: task 2
3: task 3
4: task 4
Please select an option:
```

Figure 3: easier way of doing this task

Task4: Students exam result (20 Points)

Students in a class took a multiple-choice exam where the choices range between A to D. The correct answer is as follow:

ABDCABDA

Note: You must use ~~Class~~ or **struct** for this task.

The information of students is name followed by the answers.

Struct or class can be used to get name and answers (can be a vector, string or ...).

You need to ask user to enter name and grades as an input.

You can ask user how many students information you want to get and loop through it. The user then can enter names and the scores.

Examples of how the points can be evaluated:

John	ABDCABDA	8 points	// John got 8 points
Sara	ABCCABBB	5 points	// this student got 5 answers right
Laila	BACCABCC	3 points	// this student got 3 answers right
Bob	ABDCABDC	7 points	// this student got 7 answers right

And so on;

And finally, print out the result similar way as shown below, from the highest score to lowest score:

Student name	score
John	8
Bob	7
Sara	5
Laila	3
...	

Number of students: 4

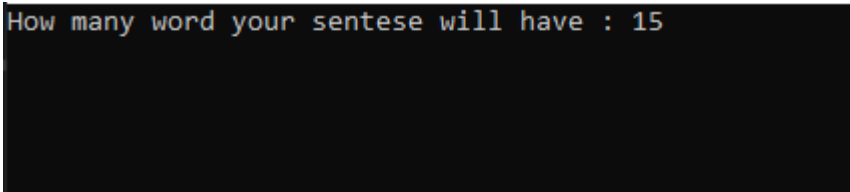
Average score: 5.75

Max score: 8

Min score: 3

Task 5: Typing a sentence using only arrow keys or AWSK keys (40 points)

In this task, you need to ask the user “How many characters they want in a sentence?”. This means you need to create a **dynamic** array to save individual characters.



```
How many word your sentese will have : 15
```

Figure 4: example shows that user wants to write a sentence and needs 15-character words for it

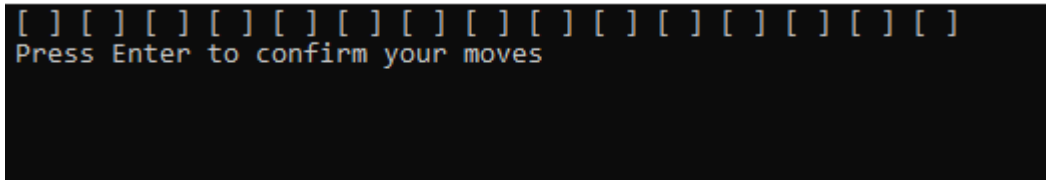


Figure 5 This shows 15 spaces allocated dynamically for the user

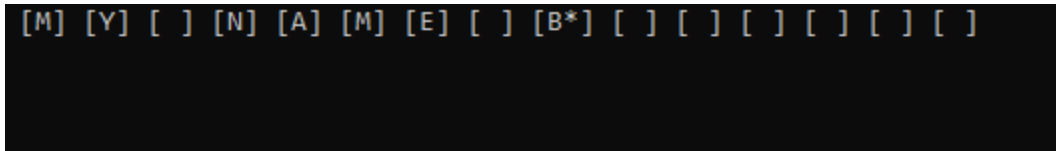


Figure 6 this shows the user moving right and left using A and D or left and right keys

In figure 6, you can see that using arrow keys, the user can go left or right. '*' indicates where the user currently is. There is empty [], which means the user pressed spacebar here to create a blank char.

When the user is done with this sentence, the user presses the enter key, and print the whole sentences on screen.