C programming basics

Lecture 5

Schedule

- Switch - case
- #define
- enums
- practice

Any questions ?

Homework solving

Questions ?

Switch-case

Problem

- Make a function that is printing a description about a mark that a student has received. Use the provided data to print the right message:
  2 - Poor :(
  3 - Average, just above the minimum
  4 - Good, need more studying and practice
  5 - Very good, a bit more to the top
  6 - Excellent, keep the spirit

switch-case

```
void printMarkDescription(int receivedMark){
    switch (receivedMark) {
        case 2:
            printf("Poor :(");
            break;
        case 3:
            printf("Average, just above the minimum");
            break;
        case 4:
            printf("Good, need more studying and practice ");
            break;
        case 5:
            printf("Very good, a bit more to the top");
            break;
        case 6:
            printf("Excellent, keep the spirit");
        default:
            printf("invalid input");
            break;
    }
}
```

switch-case

- Switch-case statements are used in programing as exclusive functions that check if a variable is exact value. C switch-case is falling though switch-case, that means that if you do not put break at the end of the statements, the next one will also be considered as true. See next slides.

switch case

```
switch (variable) {
        case 2: // if variable == 2
            //execute some code
            break;
        case 3: // else if variable == 3
            //execute other code
            break;
        default: // else
            //execute third code
            break;
    }
```

switch case - falling though

- Falling though

```
switch (variable) {
        case 2: // if variable == 2
                break;
        case 3:
        case 4: // if variable == 3 || variable == 4
                //execute something
                break;
        default: // else
                break;
}
```

Switch-case

- Make a program that is incrementing a variable with number based on user input using the following data:
  If user inputs - increments with :
  1 - 5
  2 - 9
  3 - 18
  4 - 34
  5 - 59
  6 - 95

Switch-case

- Fall though example

Questions ?

Task

- Car garage offers you a great amount of money in order to display information about the models of the Honda vehicles. The user must input a number and then the program must display the model's make and the years when it has been manufacturing.
  Example:
  User inputs 1, that stands for Honda Civic
  MK1 - 1972 to 1979
  MK2 - 1979 to 1983
  MK3 - 1983 to 1987
  MK4 - 1987 to 1991
  MK5 - 1991 to 1995
  MK6 - 1995 to 2000
  MK7 - 2000 to 2005
  MK8 - 2005 to 2011
  MK9 - 2011 to 2015
  MK10 - 2015 to present
  Input 2 and 3 stands for Honda Accord and Honda Jazz. Input their data.

Questions ?

#define

#define

- Do you see a problem in the other task. How you can be always sure that you will never mistake the numbers 1, 2, 3, etc. and display wrong data ?

#define

- In C #define is used to make an alias of an integer number. That means that if you want to make the number 1 to stand for model Civic, number 2 to stand for Accord and number 3 to stand for Jazz you will do it like this:

```
#define CIVIC 1
#define ACCORD 2
#define JAZZ 3
```

#define

- Then my switch case will be:

```
void printCardData(int userInputNumber){
    switch (userInputNumber) {
        case CIVIC:
            // print civic data
            break;
        case ACCORD:
            // print accord data
            break;
        case JAZZ:
            //print jazz data
            break;
        default:
            // incorrect input
            break;
    }
}
```

#define

- Commonly #define is used for return results of a program. For example if there is an error for the user input you program may finish with FAIL_WRONG_USER_INPUT, which must have been #define-d. That way the debugging is easier.

#define

- #define is so-called preprocessor instruction, because all the #define statements in you program are replaced with their corresponding numbers before compilation.

#define

- Common mistakes :
  - #define statements doesn’t end with ‘;’ !
  - #define statements can be redefined, so care what names do you use

Questions

Task

- Error handling:
  Make a check if user input a correct number. The correct number is dividable by 3, 5, 8. If some of the conditions is not met, return #define error as end of the program (in int main return different value than 0). If all the conditions are met, return 0 (#define 0 as SUCCESS)

Enums

Enums

- Let’s get back to the task with the garage and cars information. Do you think that is would be better if we encapsulate the different car models into a variable type ?

Enums

Alias

Value

```
typedef enum {
    Civic = 0,
    Accord = 1,
    Jazz = 2,
} Cars;
```

Name of variable type

Enums

- Enums are used to encapsulate a range of variables that have something in common. For example you can encapsulate the marks in school, different car manufacturers, etc.

- By making them a different type, you are making sure that there will be no mistakes when they are being used in the code.

Enums

- How to use an enum:

```
void printCarsData(Cars selectedCar){
    switch (selectedCar) {
        case Civic:
            // print civic data
            break;
        case Accord:
            // print accord data
            break;
        case Jazz:
            //print jazz data
            break;
    }
}
```

Task

- A school teacher needs software to easily calculate the average mark of the class. The class consists of 15 students, and the subjects are 6. The data is inputed for each student individually in format “%d,%d,%d,%d,%d,%d”, the teacher taps enter and the program requests data for the next student. At the end the program calculates the average mark for each student individually and presents data with it’s average mark alias (eg. if the average is 5.87, the program displays Excellent). At the end the program displays the average mark for the class as float number.

- Notes : Mind that 5.50 is Excellent, 5.49 is very good (;

Homework 1

- The C Programing basics lecturer needs help in the examination process. The exam consists of test that has maximum of 30 points and 5 tasks with maximum of 10 points each. There are also 10 points from homework and 10 points from homework evaluation. The data is inputed in the format “$test, $task1, $task2, $task3, $task4, $task5, $homework”, $hw_evaluation” (everything is float).
  Calculate the mark of the student, based on the sum if it’s point :
  0 - 44 = 2 (Poor)
  45 - 59 = 3 (Average)
  60 - 74 = 4 (Good)
  75 - 79 = 5 (Very good)
  80 + = 6 (Excellent).
  Print a message with the alias of the mark.

Homework 2

- Use the functions from task 1 and make a program that is used for inputing the data for 10 students and calculation their average mark. It must be printed with the alias AND the numeric value.