**Computer Science** I Fall 2014

|  |  |  |
| --- | --- | --- |
| Prog. Lang. | High (1-4) | Easy (1-4) |
| C | 1 | 4 |
| C++ | 2 | 3 |
| Objective C | 3 | 2 |
| C# (sharp) | 4 | 1 |

Computer \* HW  
 \* SW -Programing language -> C

-Compiler

Compiler -> to read our program; converts our program code to machine level code  
All compilers must support – Standard C  
Not all compilers give the same output

Intergrated developed environment IDE: => text editor  
 => compiler  
 => features

Whats progming language? 9/8/2014

C (1972) created by one guy, Compiler created by other guy.  
The lowest language- binary system (1,0) => 1960- symbolic language using symbols, ormnemonics to represent the various machine language instructions.

„return 0;“ –if there is no mistake it returns back to zero (u can put any number)

When you wrote a program always have an indent!!! Its mandatory!!!

1 indentation=3 spaces=1 tab

For each new block=indent!  
Whats a block?  
- Region between { }

Programmer== Text Editor (source file, myProgram.c) (c is just extension for simplicity)

Compiler== Object (into machine language)

Linker==Library (Executable) (Links all other sources from outside)

Runner== (screen), myProgram.exe

Creating and Running Programs  
Step 1: Use a TEXT EDITOR to write programs  
 After we complete the programs we save our file to disk

Step2 2: Use a compiler to translate program code to machinary language (preprocessor=>compiling)

Step 3: Linker assembles all required functions to run the program from the library  
 C program code is made up of C-key-words and predefined C-funtions.

Step 4: When we run the program, is is executed by the operating system.

C is a procedural programming language (not object oriented language).  
 > a series of computational steps are carried out one by ne.  
Developed in 1972 by Dennis Ritchie  
C89, C95, C99 (standard C)  
A C program is made of one or more functions.  
 > what function is the starting point?

„main function“ –starting point Preprocessor Directives  
 Global Declarations  
 int main (void)  
 {  
 Local Declaration  
 Statements  
 } // main  
 Other functions are required.

# include <stdio.h> - to include library/include standard input/output

Library- set of functions

Main.c (source code)  
Main.o (object file created by Compiler)  
Makefile.win (Makefiles are a simple way to organize code compilation, smth special in IDE)  
Project2 (Linker takes the object file (main.o) and creates an executable file (Project2)

Black window- result

\*\*\*\*hexatizm numbers

Write reminders into your source code: reminders=comments (they are usually added with the purpose of making the course code easier to understand)  
Compiler usually ignores the comments

Comment: // for single line-single comment  
 /\* \*/ multiple lines-multiple comments

„printf“ is just a function to display the calculation or action done on the screen.

Identifiers: (name of function) One feature present in all computer languages is the identifier. It allows us to name data and other objects in the program. Ech identifier object in the computer is stored at a unique address.

RULES FOR INDENTIFIER  
First character must be alphabetic character or underscore. Int myStudentId; Int my\_student\_d  
Must consist only alphabetic characters, digital or underscore.  
First 63 character of an identifier are significant.  
Cannot duplicate a keyword. (Upper-lower case sensitive)

[iPhone 6 & iPhone6+ = iOS 8 and swing (new Mobile Program introduced)]

C----C++----Java; From C++ & Java----C#; From C----Obj C 9/10/2014

C-----process data –define functions  
 -----store data –declare variable

* Main goal- to process data to be able to store the data!!! (Java functions are called methods, C++ functions, Obj C action)

Type variable Name;   
 2 integer-> number= data types   
 a letter-> string= data types 3.14🡪 float/double  
integer🡪short—2 bytes  
   
 🡪int —4 bytes

How much it is?  
How much we need?  
How much memory it requires?

🡪long—8 bytes

* Variables:

are named memory locations that have atype, such as integer or character, which is inherited from their type. The type determines the values that a variable may contain and the operations that may be used with its values.

C Types  
Void Integral Floating-point Derived  
 -- Boolen -- Real  
 -- Character -- Imaginary  
 -- Integer -- Complex

Sizeof (short) sizeof (int) sizeof (long) sizeof (long long) Predefined function sizeof

Homework #1:

Printf(„int type=%d\n“, sizeof(int));

1. Do research and find all out data types in C
2. Print size info for all data types (DUE FRIDAY, 12TH)
3. Find the library where function sizeof is defined

\n new line  
%d format for integer  
„int type=%d\n“ format string (just text for compiler)  
sizeof function  
(int) function argument 🡪 compiler will return the size of int (hackathon.com)

To build something we need; plan & materials (algorithm & data structure)

„What the schools dont teach?“ video to check 9/15/2014

C---- data----variable  
 process-----function

Variable---- declare  
type variableName;   
integer; short int  
 int  
 long int  
 long long int

#include <stdio.h>  
int main (void)  
{  
 //declare  
 int number; // you must initialize the variable before you use it

Init---{ declare  
 { inside the program  
 printf(„number=%d“, number); ---this will print something   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

.... FIND WHATS THE DIFFERENCE???  
int number=2  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Int number;  
number=2;

Input/output  
printf(.....) # of arguments can change  
printf(„Hi!“); easy way of using it (if we dont put \n the new printf function **will continue where we stopped)**

Instead of newline= \(backslash) n backslash=operator **(inside the text=message to compiler: going to print something special, focus on the right side of back slash)**

('\\') to print just backslash

**\, %** (to print number) **special characters**  
**d= to precise the number** (d=decimal integer)

**Int number=15;**(2nd part) **int total;  
total=number\*2;**

Whats happening in the memory while we declare the variable: it is going to make a space, storage for that variable!!! // variable name (number), memory address (address in memory), value of variable (30)//

**= ---- assignment operator** (left\_value=right\_value; left\_value<=right\_value) **(Calculate the right side and then assign it to the left value on the left side (which is usually a single number))**

Number=number+1  
left side right side  
1. calculate the right side  
15+1=16  
2. assign the right side to the left side  
the value of left side, the value of number is 16  
number<=16 **Compiler never looks the left side until you dont calculate the right side (we first assing the value to the numbers on the right side)**

..... int x=10;   
 int result;  
 result=x\*x;

.... int x=10, result;  
 result;  
 result=x\*x;

Fastes: .... int x=10;

Printf('%d', x\*x);

**The return value of main= integer**

**Scanf function; it scans the default input device, scans everything we type on our keyboard ex. 3+4= scans just as a three characters**!!!  
Hi!(stream of input) (enter)

Scanf('%d', x); (needs the memory address of the variable not the name)

Variable: name (printf), memory address (scanf), value (many functions will need it)  
**&x Variable Name=memory address**(we use the name to access the value)   
**&= to access the memory address**

**int x, result;**

**// get data from the user**

**scanf("%d", &x); // go to default input device**

**result=x\*x;**

**printf("%d", result);**

**return 0;**

Memory address: #1000 (random number)  
Name of variable: x  
Value of variable: 10

Memory address: #1004  
Name of variable: Result  
Value of variable:100

Scanf()--- go to default input device  
%d (inside the scanf)--- convert input system (i.e, set of characters) to integer  
 &x--- store the value of integer in x (memory address of x is #1000)

\*Power funtions for exponents and loops.\*

# include <stdio.h>

int main (void)

{

Int number1, int number2;

Float average;

average=(number1+number2)/2;

printf('Please enter the first integer:');

scanf('%d', &number1);

printf('You entered %d\n', number1);

printf('Please enter the second integer:');

scanf('%d', number2);

printf('You entered %d\n', number2);  
 printf('Average=%f', average)

return 0;

}

Int--- because we have return 0, we are entering integer and expecting integer; for example if we put float then it is going to be return 0.1;

What does int mean--- just that we are entering and waiting from program  
What does main mean--- main function; just name   
What does void mean--- void simply is 0 (program can run without it)

Int main (void) and return are connected, what we are declaring we are expecting to return to that point

**Review**

System requirements---analysis---design---code---system test----maintenance   
C compiler: preprocessor  
 actual compiler

Preprocessor directives  
Global declarations (all variables outside of block/s are called global variables); all functions share global variables

Int main (void)  
{   
 local declarations ------------block (all variables inside block are called local variables)  
 statements  
} main

Other functions as required

Compiler does not recognize comments, doesnt matter how many comments you include in your programi its not gonna slow down your program.

In C even the program is functio; thats why we have return 0; compiler calls(involving) back main function;

Int main ---- we define the function, we dont call it  
In C if we wanna call the funtion we have to define it, involve it and state arguments.

A variable has three properties: name (identifier)   
 memory address  
 value

C---Code: C keywords  
 predefined functions  
 identifiers--- we define indentifier while we cannot define others

Types of variables(to store the data): void type (does not take any argument) void number; its nothing  
 integral type (boolean (true 1-false0), character, integer  
 floating point types

ASCII Table; every letter has a corresponding

} number composed of 0s and 1s

String is a set of a chars, under the ““

#include <stdio.h>

// define function, in C everything is a function, even the program is a function

// starting point is the main function

int main (void)

{

char letter;

letter='c';

prinf("letter=%c", letter);

return 0;

if we put %d instead of c, our output is going to be number 99 (ascii code) (binary system) because characters are stored in numbers.

#include <stdio.h>

// define function, in C everything is a function, even the program is a function

// starting point is the main function

int main (void)

{

char letter;

letter='c';

prinf("letter=%d", letter+1);

return 0;

}

the output will be 99+1=100!!!

#include <stdio.h>

// define function, in C everything is a function, even the program is a function

// starting point is the main function

int main (void)

{

char letter;

letter='b';

prinf("letter=%c", letter+1);

return 0;

}

The output will be c (98+1=99=c)

#include <stdio.h>

// define function, in C everything is a function, even the program is a function

// starting point is the main function

int main (void)

{

char letter;

letter='b';

prinf("letter=%c", letter-1);

return 0;  
}

The output will be a (98-1=97=a)

Computer does everything based on numbers.

#include <stdio.h>

// define function, in C everything is a function, even the program is a function

// starting point is the main function

int main (void)

{

char letter;

letter='b';

prinf("letter=%d", 'z'-'a');

return 0;

}

The result will be 25= z (26th letter)-a(1st letter)=26-1=25

#include <stdio.h>

// define function, in C everything is a function, even the program is a function

// starting point is the main function

int main (void)

{

prinf("character=%c", 94);

return 0;

}

The output will be one of the characthers (they are stored as integers)

Short int🡪 2 bytes

1byte=8 bits (0 or 1)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

1 byte = 8 bits (for every box we can store either 0 or 1)

Lab 2:

Int cel;  
float far;

Printf(“Give .... in Celsius“);

Scanf(“%d“, &cel);

Far=1.8\*cel+32;

9/24/2014

1 byte=8 bits  
2^8=256  
0 1 2 3 4 5... 255 (numbers can be stored in 1 byte)  
256--- 128numbers <0 (-1,...-128) 128numbers>0 or equal to 0 (0,1,2...127)   
1byte=8bits, base=2

**The smallest integer number that we can store with b bytes:**

**-2^b-1**

**The biggest integer number that we can store with b bytes:**

**(2^b-1)-1**

#include <stdio.h>

Int main (void)

{

Char variable;  
 printf(“size=%d“, sizeof(char));

Return 0;  
}

#include <stdio.h>

int main (void)

{

char variable;

variable=-129;

printf("variable=%d", variable);

return 0;

} // output is going to be 127 !!!

First bit is for sign and the rest are for values ! So if it is zero it is positive, and if it 1 is it negative.  
In C 10000000 =-128

SSN whats the biggest value we can put for it, which int type we use to save it, int or long int which are 4 bytes are enough to use it.

ACSII Character Symbolic Name  
null character '\0'  
alert (bell) '\a'  
backspace '\b'  
horizontal tab '\t'  
vertical tab '\v'  
form feed '\f'  
carriage return '\r'  
single quote '\''

Computers use binary system while we use decimal numbers !!!

In binary system= from right to left the power is inscreasing !   
for example:

1011 = from right to left

1x2^0+1x2^1+0x2^2+1x2^3 2 is base

For example 6 in binary system; is 110= 0x2^0+1x2^1+1x2^2=0+2+4=6

ASCII Library contains 256 characters

65 to binary 1000001

Any binary number starting with 1 the number is odd, if it starts with 0 it is even!!!

In binary system 0 means false and 1 means true, it means anything else except zero is true. Bool-Bollean

9/29/2014

Constants  
-are data values that cannot be changed during the execution of a program. Like variables, constants have a type. In this section, we discuss Boolean, character, integer, real, complex, and string constants.   
-named 🡪 const  
-literal 🡪 'a' not named one!

Depending on what we wrote, in the memory it is stored as a variable or a constant; int a=54; int type, a name and 54 value---- then memory address for 54, it doesnt have a name and value 54, and this is our literal constant!

#include <stdio.h>

int main (void)

{

int a=54; // compiler makes a box for every constant and variable separately, first for a, then 54, then 2, then a\*2, then give us output, and the compiler will access those variables using the memory address because they dont have names [\*(#memory address)]

a=a\*2;

printf("%d", a);

return 0;

} result is going to be 108

Compiler doesnt know anything until we dont declare types of our variables, if we dont declare compiler will assume that it is integer.

36789567987--- compiler will give it memory address, it doesnt have a name, literal constant, but it will not be able to store it coz its value is bigger than the value can be stored for integer, actually compiler will store it as an integer coz we didnt not declare it, and everything we dont declare compiler stores it as an integer.

367895667987L is notation to give a sign to compiler that the number we wanna store is long integer.

#include <stdio.h>

int main (void)

{

printf("Largest int value %d", 2147483647+1);

return 0;

} // its going to print the higest number with a minus sign= -2147483648

Format specifier for long long integer= lld% , for integer=d%, for long integer=ld%

#include <stdio.h>

int main (void)

{

printf("Largest int value %lld", 2147483647LL+1); // purple part to store in memory, and yellow to display for users

return 0;

}

3.14f and 3.14L are different number for memory storage, memory for them is different (f-float-4bytes, l-long double-12 bytes)

#include <stdio.h>

int main (void)

{

printf("float :%d\n", sizeof(3.14f));

printf("long double :%d", sizeof(3.14L));

return 0;

}

Use keyword const for declearing the constant; const type VARIABLE NAME=value

In a function; Local constant; out of it is global

Purpose of declearing the const; in long programs instead of writing long numbers just write its notation (for example in our program we have to write PI for 20 000 times, so instead of writing 3.14 20 000 times, we just write PI., another thing when we wanna change the number, if we declared it as a constant it si going to be easy, we change it in onle single line where we declared it instead of going to 20 000 lines and change it one by one)

Const long double PI=3.1414L; // L coz compiler looks first at the right side.

Int main ()

10/1/2014

ASCII Table;

Char letter='a';  
printf(„%d“, letter); ---97  
printf(„%c“, letter); ----a

An expression always reduces to a single value. a=2\*b-c+12; compiler first calculates the right side and then assign that value to a, left hand side.

Compiler Questions for exam !!!

Expression categories; primary value itself X, postfix X----, prefix ------X, unary X, binary X,Y, ternary X,Y,Z (conditional operator)

Postfix Expression  
Operator Operand  
\*,+,-,/,++ Variable

(a++) (compiler does know that the value will be written at 'a') has the same effect as (a=a+1) (just compiler doesnt know where will it store the value of a+1)

The operand in a postfix must be a variable.

#include <stdio.h>  
int main ()  
{

Int a; // local declaration  
 a=4; // statements  
 printf(„value of a: %d\n“, a);  
 printf(„value of a++: %d\n“, a++); //calculate the result, but dont update the value in memory, after executing it then update it as well  
 printf(„new value of a:%d\n“, a);  
 return

}

Output: 4,4,5 explanation; a++; 1. a+1  
 2. a🡨 a+1 do the calculation, use it/execute, and then assign it after using the value.

++a -🡪 a=a+1

If ++ is after the operand, as in a++, the increment (increase) takes place after the expression is evaluated.

If ++ is before the operand, as in ++a, the increment(increase) takes place before the expression is evaluated.

Unary Operator, it means there is only one operand. (sizeof(int), sifeof -25.58)

(-) effects just single variable !!!; in C two +, sumation and sign + (part of number)

Binary Operation, there r two operands and one oberator.

10\*3 evaluates to 30  
true \*4 evaluates to 4  
'A' \*2 evaluates to 130=2\*65  
22.3\*2 evaluates to 44.6  
(2+3\*I)\*(1+2\*I) evaluates to -4+7\*I

% operator for remainder 10%3 evaluates to 1

Loop for repeating the same operation many time!!!

10/6/2014

#include <stdio.h> //ask for grade (a, b,c,d, and e) and then print if !!!

int main (void)

{

char grade;

printf("Please enter your grade.\n");

scanf("%c", &grade); // first part string, the second part memory address of a variable

// variable; mem.address, name & value

printf("You entered %c.", grade);

return 0;

}

#include <stdio.h>

int main (void)

{

char grade1,grade2;

printf("Please enter your grades.\n");

scanf("%c%c", &grade1,grade2); // first part string, the second part memory address of a variable

// variable; mem.address, name & value

printf("You entered %c and %c.", grade1, grade2);

return 0;

}

**What will be the output of this code???**

**#include <stdio.h>**

**int main (void)**

**{**

**int a=2, b=3, c=4;**

**a++;**

**b=c\*a;**

**c=b/c;**

**printf("%d", c);**

**return 0;**

**} //The output will be 3.**

**#include <stdio.h>**

**int main (void)**

**{**

**int a=2,b=3,c=4;**

**b=c\*(a++);**

**printf("a=%d, b=%d", a,b);**

**return 0;**

**} // the output will be a=3, b=8 ; because we have postfix a++, then we have b=c\*a++ ; 4\*2=8**

A++= calculate then update the value  
++A=update then calculate the value

10/15/2014

Functiions!!!

In C, everything is a function.  
Why we need functions in a program?

If everything has been in a main function with 4000 line it would be confusing and hard to find errors and to work with that code, also it would be hard to reuse some of the part of code.

Function, reason why we have funtions; divide problem into smaller parts; write a function for every part, if there is error it would be easier to find which part is wrong because it is going to show use under which function is error, also it is easier to reuse some parts if we use functions, because we can include functions in library.

Function:

Easy to understand  
Easy debug  
Code re-use

-type-returned  
-name  
-arguments  
-result-what the function will return

typeReturned functionName (argument list)  
{  
return result;  
}

arguement lis: type of it and name of it (type1 argName1, type2 argName2, ...)

How to call or invoke a function: functionName (paramenteres)  
parameter=argument

What is the difference between macro and function????   
Macro is not function, it is shorter than as function !!!

Kernel- one pf the packages is library

10/20/2014

If-else, for, and, and or

Input-exam grade, float  
Output-letter grade, char

If( )- logical comparison true or false, it if true one output if it is false another output

If( logical expression)  
{  
}  
else  
{  
}

If else (multiples options) & for (when we have to use function multiple times!)

For (init. Counter; chech condition; update counter)  
counter=0, counter<=100, counter++  
{  
}

|| or

&& and

10/22/2014

Int counter  
Int times=0;  
for(counter=-3; counter<0; counter++)  
{  
Times++  
}

Times counter counter  
0 -3 0  
times++ counter++ counter<0  
1 -2 -2  
times++ counter++ counter<0  
2 -1 -2  
times++ counter++ counter<0  
3 0 0

Int result=-2 (case1)  
for(counter=5; counter<-3; counter++)  
{  
 result=result+2\*counter  
}

Result counter counter<-3  
-2 5 8  
DO NOT EXECUTE BECAUSE IT IS FALSE  
the value of result is -2, and value of counter is 5 (We update counter after we execute for body)

Int result=2 (case2)  
for(counter=5; counter<-3; counter--)  
{  
 result=result+2\*counter  
}

Result counter counter<-3  
2 5 12 FALSE (never executes)

Int result=2 (case3)  
for(counter=5; counter>-3; counter++)  
{  
 result=result+2\*counter  
}

Result counter counter>-3 INFINITE LOOP  
-2 5 8 True  
8 6 20 True  
20 7 34 True  
34 8 50 True  
50 9 68 True

General structure for loops:  
 1. init counter  
 2. check condition  
 3. T execute for body  
 4. F do not execute for body (if it is false skip for(not the whole program)  
 5. update counter (after it check condition)

\*

\*\*

\*\*\*