C++ Cheat Sheet

Class: CSCI 1300

General Reminders

code	*description*
# include ”myfile.h”	Include file.
# include<cassert>	Include assert library.
assert(boolean);	Throw an error if the boolean is true.
rand()	Random int (# include<cstdlib>).
abs(x)	Returns $\|x\|$ (# include<cstdlib>).
int(var)	Convert a primitive data type var to int.
int* myInt;	* means myInt work form a pointer.
&var	Get mem addresse and pass by var by ref.
void myF() const	read only fonction

Strings

code	*description*
str[i]	Get or set the char at the index i.
str.length()	Return the number of characters.
str.substr(a,b)	Returns the substring from a to b.
str.find(subStr)	Retrun the start index of the substring
str.replace(i,l,str)	Replace substring from i with str
stoi(str)	Convert a string to int(# include<string>).

Arrays

0	1	...	n
”Max”	”Tom”	...	arr[n]

This table illustrate the structure of an array of strings. Considering that n is equal to the number of element minus one. Arrays are a static data type.

code	*description*
int arr[4];	Create a array of int and with 4 element.
int arr[4]={6,3};	
arr[i]	Get or set the element at the index i.

Vectors

code	*description*
# include<vector>	Include vector library.
vector<type> V;	Instantiate a vector.
vector<type> V(size);	Instantiate a vector from Array obj.
vector<type> V{6,3,3};	Instantiate a vector from Array.
V = vector<type>();	Re-instantiate V
V.at(Index)	Returns the element at index i.
V.size()	Return the number of elements.
V.push_back(Value)	Add the new element at the end.
V.pop_back()	Remove the last element.
V.clear()	Empty the vector.
V.insert(Index, Value)	Insert element at i.

Structures

```cpp
struct myStruct {
    string param1;    // atribute 1
    double param2;   // atribute 2
}s1, s2;            // myStruct instances
```

code	*description*
myStruct Obj;	instantiate structure object.
Obj.param1	Access param1 of Obj.

Streams

code	*description*
# include<fstream>	Include stream library.
# include<sstream>	Include string stream library.
ifstream fin;	Instantiate a input stream.
ofstream fout;	Instantiate a output stream.
stringstream s(myStr);	Instantiate a string stream.
myS.open(”file.txt”)	Open txt file whith the stream.
myS.close()	Close the stream file.
getline(fin, line)	Get the next line from fin.
fout<<”hello”	Output in stream ”helloWorld”.
fin>>var	Input from stream to var.
<<setprecision(n)<<	Set decimal points (#include<iomanip>)
<<setw(n)<<	Establishes a print field of n spaces.
<<fixed<<	Display floating point numbers in fixed. point notation.
<<showpoint<< <<noshowpoint<<	Enables or disables the unconditional inclusion of the decimal point character in floating-point output.
<<left<<	output the string on the left.
<<right<<	output the string on the right.

clear buffer

The buffer must be cleared after after getting an input from a stream if you input and output in the same file at the same time.

```cpp
if(cin.fail() == true) {
    cout << "cin failed state";
    cin.clear();
    cin.ignore(1000, '\n');
}
```

cmath

code	*description*
# include<cmath>	Include cmath library.
sqrt(x)	Square root of x.
pow(x, y)	x raised to the power y.
abs(x)	Absolute value overloads.
floor(x)	Greatest integer $\leq x$.
ceil(x)	Smallest integer $\geq x$.
fmod(x, y)	Floating-point remainder of x/y.
sin(x)	Sine of x in radians.
cos(x)	Cosine of x in radians.
tan(x)	Tangent of x in radians.

Object Oriented Programing(OOP)

```cpp
class myClasses {
    private:
        int param1;
    public:
        int param2;
        myClasses(int p1, int p2){ // constructor
            param1 = p1;
            param2 = p2;
        }
        myClasses(){ // default constructor
            param1 = -1;
        }

        string getParam1() { //getter
            return param1;
        }

        void setParam1(int p1) { // setter
            param1 = p;
        }
};
```

code	*description*
myClasses myObj(3,5);	Instantiate an myClasses type obj.
myClasses myObj;	Call the default constructor.

OOP With header file

If you use a header the file wich contain the main function must include the header file.

Header file(myHeader.h)

```
#ifndef MYCLASS_H //if no def for MyClass
#define MYCLASS_H //else

using namespace std;

class MyClass{
    public:
        MyClass(); //default constructor
        MyClass(p1, p2); //parameterized constructor
        int publicAtribute;
        void myFunction() const;
    private:
        int privAtribute;
};
#endif
```

Class file(.cpp)

```
#include <iostream>
#include  "myHeader.h"

MyClass::MyClass(){
    publicAtribute = 0;
    privAtribute = 0;
}

MyClass::MyClass(int p1, int p2){
    publicAtribute = p1;
    privAtribute = p2;
}

MyClass::void myFunction() const{
    // my code
}
```

Switch case

```
int x;
switch (x){
    case 0:
        /*Code in case 0*/
    break;
    ⋮
    case n:
        /*Code in case n*/
    break;
    default:
        /*Code if no case match*/
}
```

Important ASCII Conversions

ASCII	*int*	ASCII	*int*	ASCII	*int*	ASCII	*int*	ASCII	*int*
A	65	a	97	N	78	n	110	0	48
B	66	b	98	O	79	o	111	1	49
C	67	c	99	P	80	p	112	2	50
D	68	d	100	Q	81	q	113	3	51
E	69	e	101	R	82	n	114	4	52
F	70	f	102	S	83	s	115	5	53
G	71	g	103	T	84	t	116	6	54
H	72	h	104	U	85	u	117	7	55
I	73	i	105	V	86	v	118	8	56
J	74	j	106	W	87	w	119	9	57
K	75	k	107	X	88	x	120		
L	76	l	108	Y	89	y	121		
M	77	m	109	Z	90	z	123		