

```

110 //! Data Types (sizes/range)
111 /*
112 ! note : the number of bytes defernet depending on
113 ~ 1) the operating system 32 bit or 64 bit
114 ~ 2) type of operating system
115 % note: by default everything is signed
116 ? int: (integer)
117     # the size in Byte : 4
118     range -2147483648 to 2147483647
119     ^ it get 2 value negative and positive
120 ? Float: (floating point)
121     # the size in Byte : 4
122     range 1.17549e-38 to 3.40282e+38
123     ^ it get 2 value negative and positive
124 ? double: (double floating point)
125     # the size in Byte : 8
126     range 2.22507e-308 to 1.79769e+308
127     ^ it get 2 value negative and positive
128 ? char: (character)
129     # the size in Byte : 1
130     range -127 to 127
131 ? wchar_t: (wide character)
132     # the size in Byte : 2
133 ? bool: (boolean)
134     # the size in Byte : 1
135     range 0 (false) or 1 (true)
136 ? void: (empty)
137     # the size in Byte : 0
138 ? string:
139     # the size in Byte : 12
140
141 ! Type Modifiers:
142 ? it is used to modify some of fundimental data types (int , double , char)
143 # there are 4 type modifiers in c++
144 ~ 1) signed
145     $ int
146     * by default signed that mean yiu can use positive and negative number
147     * the range of signed int -2147483648 to 2147483647
148     * the size of signed int 4 bytes
149     ^ note : the + / - character store inside the last bit in 4 byte

```

```

150 ~ 2) unsigned
151 % use it when you want to get more space/ range without using additional byte
152 $ unsigned int
153     # it used to store positive nuber just
154     * the get last byte inside the 4 byte use it for number
155     ! so the range will be bigger from (0 to 4294967295)
156 ~ 3) short
157 $ short int / short
158     ^ it is signed short int by default
159     ^ the size of signed short int 2 bytes
160     ^ the range from -32768 to 32767
161 $ unsigned short int / short
162     ^ the size of unsigned short int 2 bytes
163     * the get last byte inside the 4 byte use it for number
164     ^ so the range will be bigger from 0 to 65536
165 ~ 4) long
166 $ long int
167     ^ it is signed long int by default
168     ^ the size of signed long int 4 bytes
169     ^ the range from -2147483648 to 2147483647
170 $ unsigned long int
171     ^ the size of unsigned long int 4 bytes
172     * the get last byte inside the 4 byte use it for number
173     ^ so the range will be bigger from 0 to 4294967295
174 ~ 5) long long
175 $ long long int
176     ^ it is signed long long int by default
177     ^ the size of signed long int 8 bytes
178     ^ the range from  $-(2^{63})$  to  $(2^{63})-1$ 
179 $ unsigned long long int
180     ^ the size of unsigned long long int 8 bytes
181     * the get last byte inside the 8 byte use it for number
182     ^ so the range will be bigger from 0 to 18446744073709551651
183 ~ 6) float
184 $ its signed and cannot be unsigned
185     ^ the size of it 4 byte
186     % the range 1175449e-38 to 340282e+38
187 ~ 7) double
188 $ its signed and cannot be unsigned
189     ^ the size of it 8 byte
190     % the range 222507e-308 to 179769e+308
191 ~ 8) long double
192 $ its signed and cannot be unsigned
193     ^ the size of it 12 byte
194     % the range 10 e-307 to 10e+308

```

```

195 ~ 9) char
196     $ signed char
197         ? its by default
198         ? the size 1 byte
199         ? the range from -127 to 127
200     $ unsigned char
201         ? the size 1 byte
202         ? the range from 0 to 255
203 ~ 10) bool
204     ^ the size of it 1 byte
205 ~ 11) string
206     ^ the size of it 12 byte
207 ~ 12) void (empty)
208     ^ the size of it 0 byte
209
210 % notes:
211     ? #1E#2 --> #1 * 10 ^ #2
212     ? if you writ number number out of range the data type it will cause an error
213     # (overflow) mean that error out of range
214     ~ in some compiler if you write wrong value like:
215         $ out of range
216         $ or assign negative value inside signed variable
217         * it will give rubbsh data
218

```

```

219 ! to get the size of data type: use sizeof(dataType) method
220     ? sizeof(bool) //1
221     ? sizeof(char) // 1
222     ? sizeof(short int) //2
223     ? sizeof(int) // 4
224     ? sizeof(long) //4
225     ? sizeof(long long) // 8
226     ? sizeof(float) // 4
227     ? sizeof(double) // 8
228

```

```

229 ! to konw the range of the data type use the dataType_MIN and dataType_MAX
230     $ cout<< "char Range : "<< CHAR_MIN <<"," << CHAR_MAX ;
231     $ cout<< "unsigned char Range : "<< 0 <<"," << UCHAR_MAX ;
232
233
234     $ cout<< "short int Range : "<< SHRT_MIN <<"," << SHRT_MAX ;
235     $ cout<< "unsigned short int Range : "<< 0 <<"," << USHRT_MAX ;
236
237     $ cout<< "int Range : "<< INT_MIN <<"," << INT_MAX ;
238     $ cout<< "unsigned int Range : "<< 0 <<"," << UINT_MAX ;
239     $ cout<< "long int Range : "<< LONG_MIN <<"," << LONG_MAX ;
240     $ cout<< "unsigned long int Range : "<< 0 <<"," << ULONG_MAX ;
241
242     $ cout<< "long long int Range : "<< LLONG_MIN <<"," << LLONG_MAX ;
243     $ cout<< "unsigned long long int Range : "<< 0 <<"," << ULLONG_MAX ;

```

```

244
245     $ cout<< "float Range : "<< FLT_MIN <<"," << FLT_MAX ;
246     $ cout<< "float (negative) Range : "<< -FLT_MIN <<"," << -FLT_MAX ;
247
248     $ cout<< "double Range : "<< DBL_MIN <<"," << DBL_MAX ;
249     $ cout<< "double (negative) Range : "<< -DBL_MIN <<"," << -DBL_MAX ;
250     $ cout<< "long double Range : "<< LDBL_MIN_10_EXP <<"," << LDBL_MAX_10_EXP ;
251
252

```