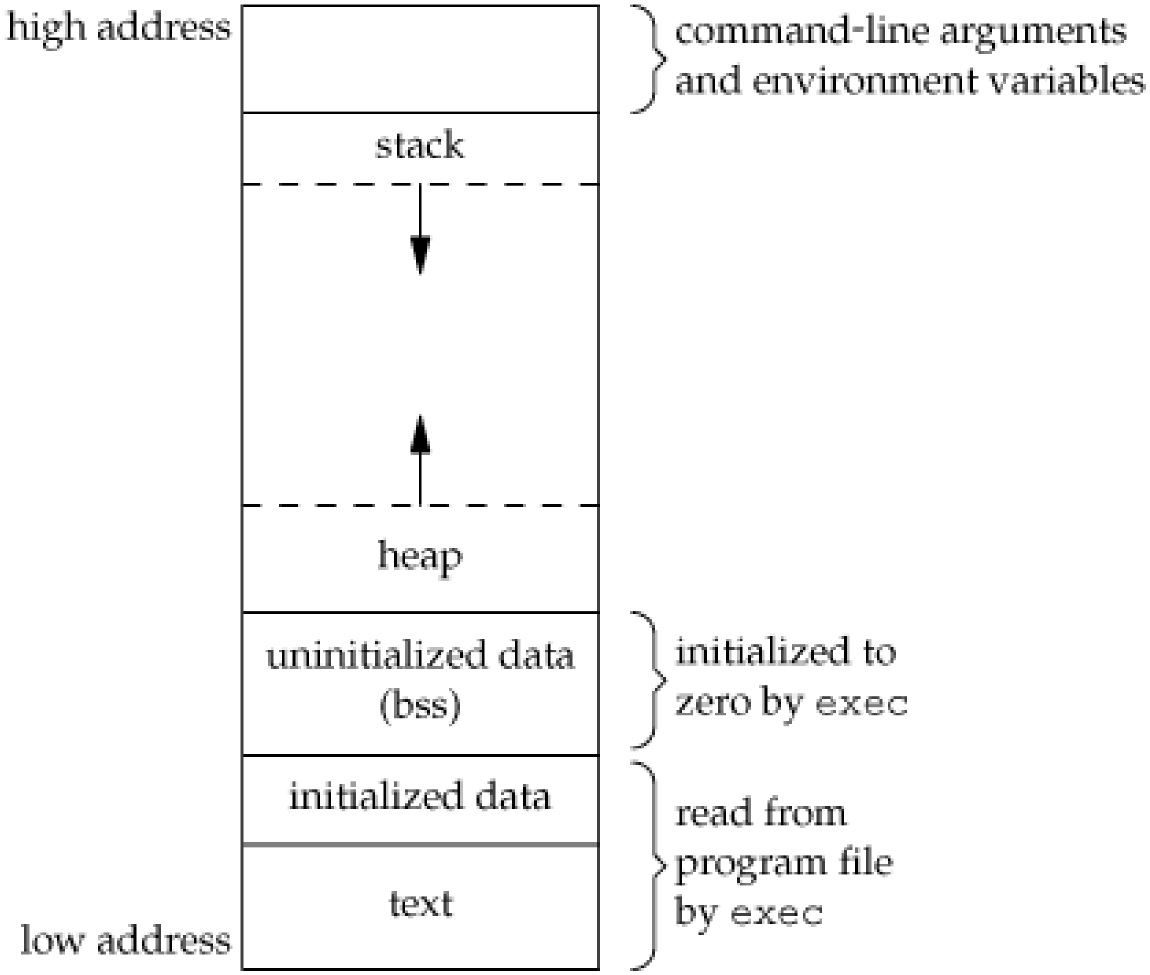
Tutorial 5: Solutions

Ravikishore Kommajosyula

Memory Layout



Source: http://www.geeksforgeeks.org/memory-layout-of-c-program/

Type casting in C++

- Static typecast:
  - Implicit conversion between types, casting through inheritance hierarchies
  - No runtime check on type
- Dynamic typecast:
  - Used for handling polymorphism
  - Cast a pointer or reference to a polymorphic type
- Re-interpret typecast:
  - Cast one type to another (most dangerous cast)

Source: http://stackoverflow.com/questions/332030/when-should-static-cast-dynamic-cast-and-reinterpret-cast-be-used

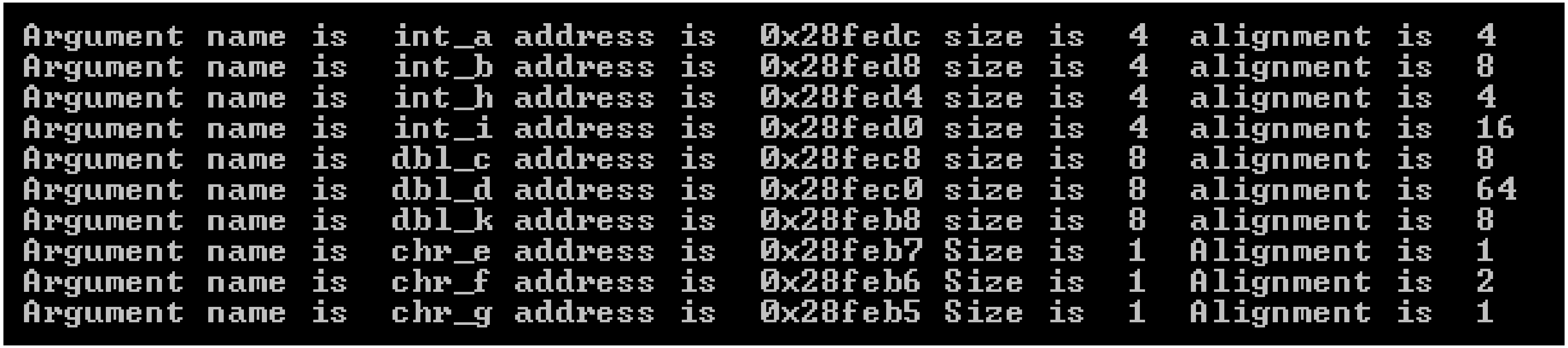
Datatype size_t

- Size_t type is a base unsigned integer type
- Size is chosen to store the maximum size of a theoretically possible array of any type
- Enables us to write portable code (32 vs 64 bit)
- Safety of normal and address arithmetic
- Performance gain by using size_t and ptrdiff_t

Source: http://www.codeproject.com/Articles/60082/About-size_t-and-ptrdiff_t

Assignment 1 in Windows

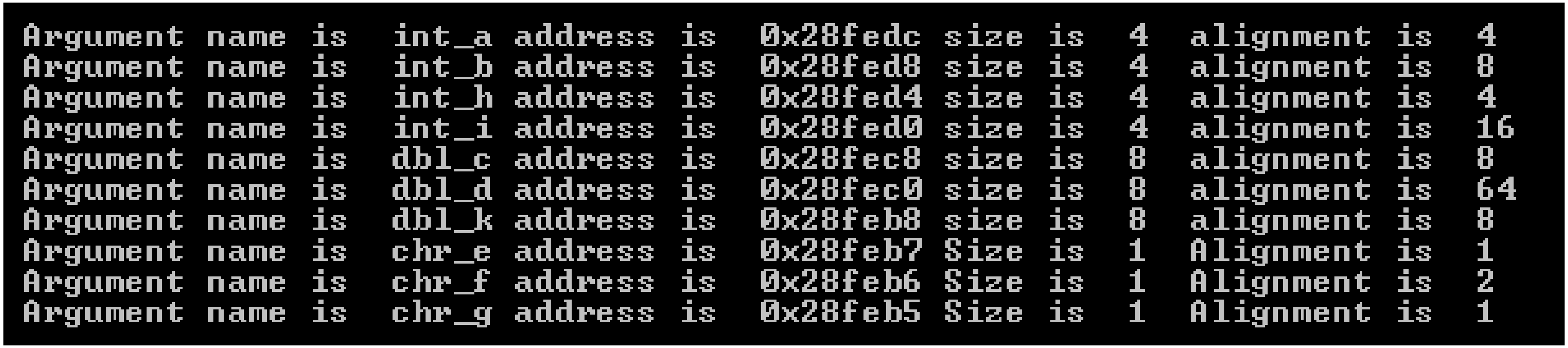
Variables declared in the right order:



```
Argument name is  int_a address is  0x28fedc size is  4  alignment is  4
Argument name is  int_b address is  0x28fed8 size is  4  alignment is  8
Argument name is  int_h address is  0x28fed4 size is  4  alignment is  4
Argument name is  int_i address is  0x28fed0 size is  4  alignment is  16
Argument name is  dbl_c address is  0x28fec8 size is  8  alignment is  8
Argument name is  dbl_d address is  0x28fec0 size is  8  alignment is  64
Argument name is  dbl_k address is  0x28feb8 size is  8  alignment is  8
Argument name is  chr_e address is  0x28feb7 Size is  1  Alignment is  1
Argument name is  chr_f address is  0x28feb6 Size is  1  Alignment is  2
Argument name is  chr_g address is  0x28feb5 Size is  1  Alignment is  1
```

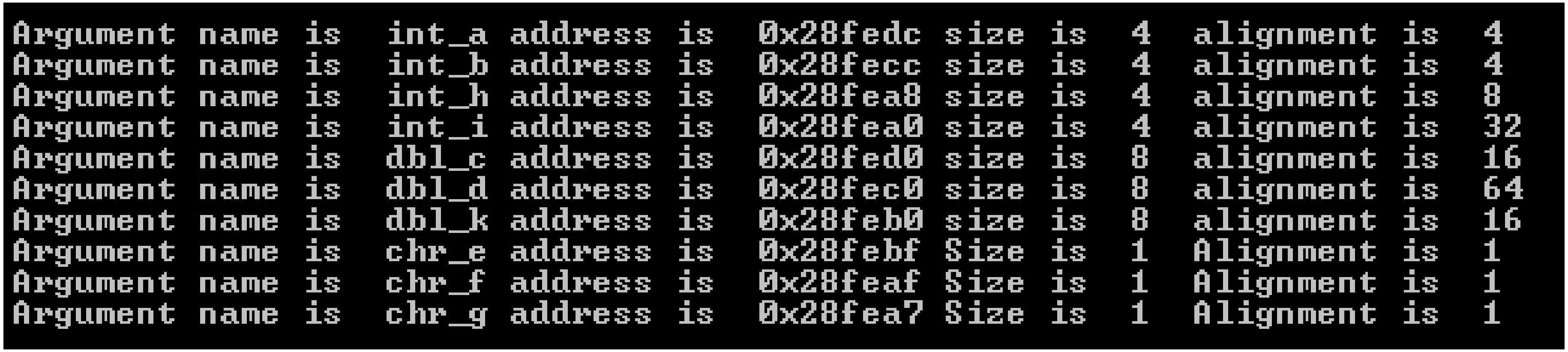
Assignment 1 in Windows

Variables declared in the right order:



```
Argument name is  int_a address is  0x28fedc size is  4  alignment is  4
Argument name is  int_b address is  0x28fed8 size is  4  alignment is  8
Argument name is  int_h address is  0x28fed4 size is  4  alignment is  4
Argument name is  int_i address is  0x28fed0 size is  4  alignment is  16
Argument name is  dbl_c address is  0x28fec8 size is  8  alignment is  8
Argument name is  dbl_d address is  0x28fec0 size is  8  alignment is  64
Argument name is  dbl_k address is  0x28feb8 size is  8  alignment is  8
Argument name is  chr_e address is  0x28feb7 Size is  1  Alignment is  1
Argument name is  chr_f address is  0x28feb6 Size is  1  Alignment is  2
Argument name is  chr_g address is  0x28feb5 Size is  1  Alignment is  1
```

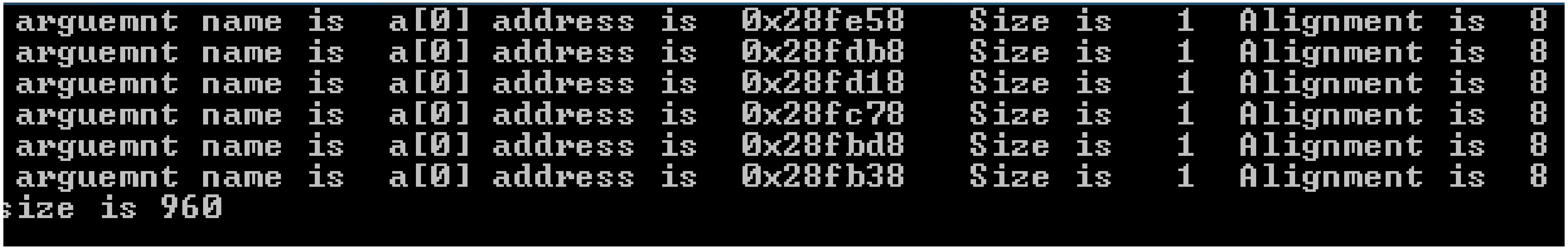
Variables declaration order Jumbled:



```
Argument name is  int_a address is  0x28fedc size is  4  alignment is  4
Argument name is  int_b address is  0x28fecc size is  4  alignment is  4
Argument name is  int_h address is  0x28fea8 size is  4  alignment is  8
Argument name is  int_i address is  0x28fea0 size is  4  alignment is  32
Argument name is  dbl_c address is  0x28fed0 size is  8  alignment is  16
Argument name is  dbl_d address is  0x28fec0 size is  8  alignment is  64
Argument name is  dbl_k address is  0x28feb0 size is  8  alignment is  16
Argument name is  chr_e address is  0x28febf Size is  1  Alignment is  1
Argument name is  chr_f address is  0x28feaf Size is  1  Alignment is  1
Argument name is  chr_g address is  0x28fea7 Size is  1  Alignment is  1
```

Assignment 2 in Windows

Output of assignment 2: (compare stack size with Ubuntu)



```
 arguemnt name is  a[0] address is  0x28fe58   Size is   1  Alignment is  8
 arguemnt name is  a[0] address is  0x28fdb8   Size is   1  Alignment is  8
 arguemnt name is  a[0] address is  0x28fd18   Size is   1  Alignment is  8
 arguemnt name is  a[0] address is  0x28fc78   Size is   1  Alignment is  8
 arguemnt name is  a[0] address is  0x28fbd8   Size is   1  Alignment is  8
 arguemnt name is  a[0] address is  0x28fb38   Size is   1  Alignment is  8
size is 960
```