C No Evil

CS 241

August 30, 2013

MP0

```
/**
 * Calculates a value (c) based on the input parameters
 * (a, b) and prints out the result.
 *
 * @param a
 *     Input parameter a.
 *
 * @param b
 *     Input parameter b.
 */
void one(const int a, const int b)
{
     int c = (a * a) + (b * b);
     cout << a << "^2 + " << b << "^2 = " << c << endl;
}
```

MP Grading

- **Execution (90%)**: Does your program produce the correct output?
  - Efficiency is not important, but it must run in a reasonable amount of time.
    - May be killed if it takes more than 10x the running time of my solution.

- **Memory (10%)**: Does `valgrind` report your program as leak and error-free?
  - Run your program with `valgrind`
  - Look at the output!

Review

`cout` →

`new` →

`int f(int &a)` →

`for (int i=0; …)` →

`sizeof(anything *)` == `____`

strings →

64-bits == ___ bytes

Program #1

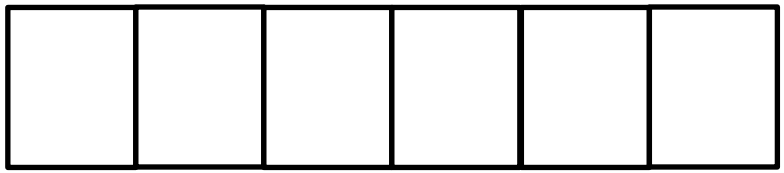
```
const int size = 10;
void main()  {
     int **values;




     for (i = 0; i < size; i++) {

          for (j = 0; j < size; j++)




}
```

Program #2a

```
void main()  {
      char *h = "Hello";
      char *w = "World";

      char *s = h + w;









      printf("%s\n", s);
}
```

C Strings

- “C Strings” are “strings” in C.
  - A “C String” is a pointer to the beginning of a sequence/array of characters.
  - The string ends when it reaches a NULL character (value: 0x00).



```
char *s = "Hello";

char *t = malloc(10 * sizeof(char));
strcpy(t, "Hello");
```

String Operations

- **char * strcpy**(char *dest, char *src)
  - Copies src to dest (will overwrite dest)
- **char * strcat**(char *dest, char *src)
  - Concatenates src onto the end of dest
- **char * strstr**(char *haystack, char *needle)
  - Finds the substring needle in haystack
- **int strcmp**(char *str1, char *str2)
  - Compares str1 and str2
    - `<0: str1 < str2`
    - `==0: str1 == str2`
    - `>0: str1 > str2`

Program #2b

```
void main()  {
      char *h = "Hello";
      char *w = "World";









      printf("%s\n", s);
}
```

Program #3

```
char * my_strcat(char *dest, char *src)  {




}
```

Pointer Arithmetic

- In C and C++, pointer arithmetic works the same way.

```
type *p = 100;
p = p + 1;    // p + 1 advances 1 sizeof(type)
```

char? **char ***? **my_struct (sizeof() == 100)**?

```
p += 6;
```

Program #4

```
int *square_ptr(int num) {
    int sq = num * num;
    return &sq;
}

void main() {
    int *sq4 = square_ptr(4);
    printf("4^2 = %d\n", sq4);
}
```

Starting a Program

- Three versions of **`main()`**:
  - **`void main()`**
  - **`int main()`**
  - **`int main(int argc, char **argv)`**
    - **argc**: Number of elements in **argv** string array.
    - **argv**:
      - **`[0]`**: Name of the process (eg: **`"./mp0"`**)
      - **`[1]`**: First command line argument
      - **`[2]`**: Second command line argument
      - …
      - **`[argc - 1]`**: Last command line argument
      - **`[argc]`**: **`"\0"`**