Computer Science 7A – Final Exam – Spring '13 Name_____

Show your work for credit. Write all responses on separate paper. Do not use an electronic computer.

1. Consider the following list of keywords in C++ (not complete.)
   `bool, break, case, char, const, continue, default, do, double, else, enum, false, float, for, if, int, long, namespace, return, short, signed, sizeof, switch, true, typedef, unsigned, using, void, while`
   a. Which of these keywords are primitive types?
   b. Which of these keywords are for building control structures?
   c. Which of these keywords is used to calculate the size of any datatype, measured in the number of bytes required to represent the type?
   d. Which of these key words allow the programmer to create a group for entities like variables and functions under a name so the global scope can be divided in "sub-scopes", each one with its own name?

2. Consider the following complete program:

```
#include <iostream>
using namespace std;

int f(int &i) {
   i = 10;
   return(5 * i);
}

int main() {
    int n = 5;
    cout << f(n) << endl;
    cout << n << "\n";
    return 0;
}
```

   a. What is the input to `f()` and what does `f()` do to its input?
   b. What is printed to the console by this program?

3. Consider the complete program shown at right.
   a. Describe the function call in the body of `main()`. What is the name of the function? What is its parameter list? What is the return type?
   b. Describe, in detail, what is happening with the assignment `s[i]=(char)(65+a%10)`.
   c. What is the purpose of this assignment: `s[i]='\0'`
   d. What is output by the program if the user enters `346609` ? To figure this out, make a table of values of `i, a, s[i]` and tabulate these values as the function is executed.
   e. This program can't produce the output ZA. Why not? How could it be modified to do so and what input would then produce ZA?

```
#include<iostream>
using namespace std;
char* int2str(int a, char s[]) {
     int i=0;
     while(a!=0) {
         s[i]=(char)(65+a%10);
         a /=  10;
         i++;
     }
     s[i]='\0';
     return s;
}

void main() {
     int a, i = 0;
     char s[100];
     cout << "Enter a: ";
     cin >> a;
     char* answer = int2str(a, s);
     cout << answer << endl;
}
```

4. Consider the code snippet:

```
int main() {
   int x = 2147483646;
   for(int k = 0; k < 5; k++)
      cout << x + k << endl;
}
```

```
2147483646
2147483647
-2147483648
-2147483647
-2147483646
```

The output of this function on a compiler that uses 4 bytes to represent an **`int`** is shown at right. Explain. What power of 2 is involved here and how?

5. The design of the code shown at right is to take a positive integer and report its divisors. The `main()` function is a driver to check that the divisors of 30 are properly reported.
   a. Fill in the conditional statement where the comment `(/*Enter code here to check if n is divisible by trial_divisor*/)`
   b. Fill in the body of main() so that it calls `print_divisors()` with an input `n=30`.
   c. What will be the output of `print_divisors(30)` ?

```
void print_divisors (int n)
{
   cout << n << endl;
   int trial_divisor;
   for (trial_divisor = n / 2;
        trial_divisor > = 1;
        --trial_divisor)
      if (/*Enter code here to check
            if n is divisible by
            trial_divisor*/)
   cout << trial_divisor << endl;
   cout << endl;
}

int main()
{
      /*Enter code here*/
}
```

6. What will the following code fragments print to the console? Why? Explain in terms of the trace of input/output and how the function is overloaded.

a.
```
int myfunc(double n) {
    return n * 2.0;
}
int myfunc(float n) {
    return n * 3.0;
}
int main() {
    cout << myfunc(3.5) << endl;
    return 0;
}
```

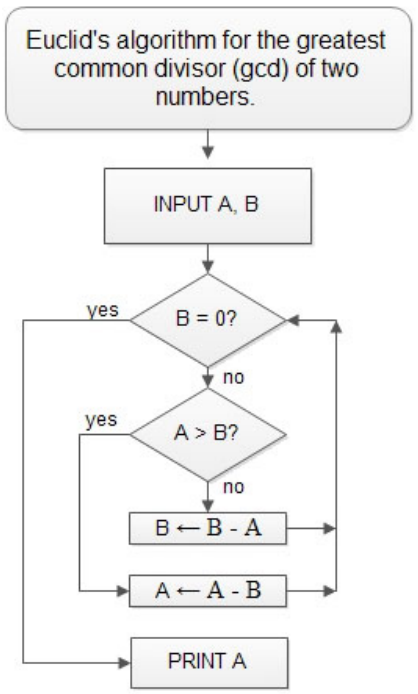
b.
```
#define SIZE 4
int sum(int v[], int n) {
   int value = 0;
   for(int i=0; i<n; ++i) {
      value += v[i]/10;
   }
   return value;
}

double sum(double v[], int n) {
   double value = 0;
    for(int i=0; i<n; ++i) {
       value += v[i]/10.;
   }
   return value;
}
int main() {
   int a1[SIZE];
   double a2[SIZE];
   for (int i = 0; i < SIZE; i++) {
      a1[i] = i*i;
      a2[i] = i*i;
   }
   cout << sum(a1, SIZE) << endl;
   cout << sum(a2, SIZE) << endl;
   return 0;
}
```

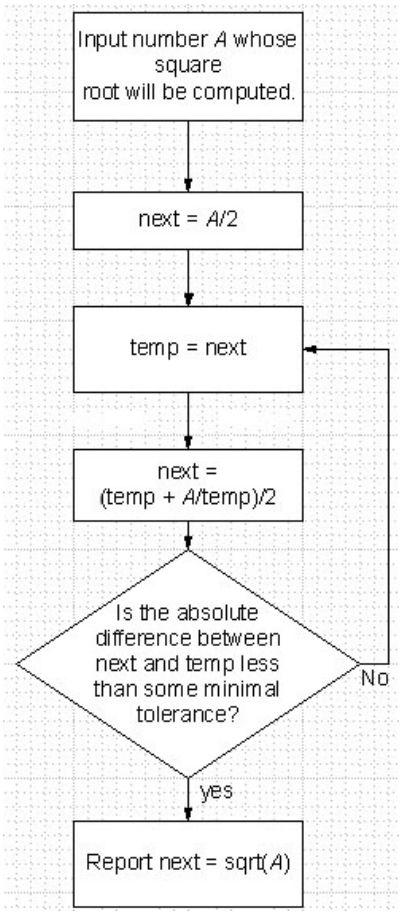
7. How much memory is reserved by each of the following allocation?
   Assume that one char requires 1 byte, an int requires 4 bytes and a double requires 8 bytes.
   a. `unsigned numbs[10];`
   b. `char word1[10];`
   c. `double matrix[10][10];`

8. Write C++ programs to implement the algorithm described by each flow chart below.

a.



b.



9. Write a function, `get_number_from_user()` meet the specificationm to meet the specification as described below.

```
/************* GET NUMBER FROM USER ***************
DESCRIPTION: This function prompts an interactive user for a positive
integer, reads the integer, and returns it in the variable passed to this
function.
PARAMETER:
n: The positive integer supplied by the user. This is a reference
parameter. It is intended that the argument passed through this parameter
will receive a new value.
RETURNS: void (no value).
ALGORITHM: The user is prompted for the positive integer. If the integer
entered by the user is zero or negative, the user will be informed that
the input is not positive and will be given another chance to enter valid
data. The function will not exit until the user has entered valid data.
*************************************************************/
```

Computer Science 7A – Final Exam Solutions – Spring '13 .

1. Consider the following list of keywords in C++ (not complete.)
   `bool, break, case, char, const, continue, default, do, double, else, enum, false, float, for, if, int, long, namespace, return, short, signed, sizeof, switch, true, typedef, unsigned, using, void, while`
   a. Which of these keywords are primitive types?
      SOLN: `bool, char, const, double, float, int, long, short, signed, unsigned, void`
   b. Which of these keywords are for building control structures?
      SOLN: `break, case, continue, default, do, while, else, false, for, if, return, switch, true,`
   c. Which of these keywords is used to calculate the size of any datatype, measured in the number of bytes required to represent the type?
      SOLN: `sizeof`
   d. Which of these key words allow the programmer to create a group for entities like variables and functions under a name so the global scope can be divided in "sub-scopes", each one with its own name? SOLN: `using, namespace,`
2. Consider the following complete program:

   a. What is the input to `f()` and what does `f()` do to its input?
      SOLN: `f` takes the address of, or reference to an `int`. `f` assigns the number 10 to the address it was passed and then returns 5 times 10, or 50.
   b. What is printed to the console by this program?
      SOLN: The output to the console is
      50
      10

```
#include <iostream>
using namespace std;

int f(int &i) {
   i = 10;
   return(5 * i);
}

int main() {
    int n = 5;
    cout << f(n) << endl;
    cout << n << "\n";
    return 0;
}
```

3. Consider the complete program shown at right.
   a. Describe the function call in the body of `main()`. What is the name of the function? What is its parameter list? What is the return type?
      SOLN: `int2str()`takes an `int` and an array of `chars` and returns a pointer to a `char`
   b. Describe, in detail, what is happening with the assignment `s[i]=(char)(65+a%10)`.
      SOLN: The remainder when `a` is divided by 10 is added to 65 and this integer is converted to an ASCII character which is then assigned to the array of characters, `s`, at offset `i`.
   c. What is the purpose of this assignment: `s[i]='\0'` SOLN: The end of string `char.`
   d. What is output by the program if the user enters `346609` ? To figure this out, make a table of values of `i, a, s[i]` and tabulate these values as the function is executed. SOLN: The integer is converted to the string "`JAGGED`"
   e. This program can't produce the output ZA. Why not? How could it be modified to do so and what input would then produce ZA? `s[i]=(char)(65+a%10)` can only produce letters `A-J.` Change that to `s[i]=(char)(65+a%26)` to get the full alphabet.

```
#include<iostream>
using namespace std;
char* int2str(int a, char s[]) {
  int i=0;
  while(a!=0) {
      s[i]=(char)(65+a%10);
      a /=  10;
      i++;
  }
  s[i]='\0';
  return s;
}

void main() {
  int a, i = 0;
  char s[100];
  cout << "Enter a: ";
  cin >> a;
  char* answer = int2str(a, s);
  cout << answer << endl;
}
```

4. Consider the code snippet:

```
int main() {
   int x = 2147483646;
   for(int k = 0; k < 5; k++)
      cout << x + k << endl;
}
```

```
2147483646
2147483647
-2147483648
-2147483647
-2147483646
```

The output of this function on a compiler that uses 4 bytes to represent an **`int`** is shown at right. Explain. What power of 2 is involved here and how?

SOLN: An `int` requires 4 bytes, which is 32 bits. 2^{31} = 2147483648 so when this value is reached the overload leads to "wrap-around" so you get –2147483648 instead of 2147483648.

5. The design of the code shown at right is to take a positive integer and report its divisors. The `main()` function is a driver to check that the divisors of 30 are properly reported.
   a. Fill in the conditional statement where the comment `(/*Enter code here to check if n is divisible by trial_divisor*/)`
      SOLN: Here's what you want there:
      `n%trial_divisor == 0`
   b. Fill in the body of main() so that it calls `print_divisors()` with an input `n=30`. SOLN: Here's the body:

```
cout << "\nprint_divisors(30) = "
     << " ";
print_divisors(30);
```

   c. What will be the output of `print_divisors(30)` ?
      SOLN:
      `print_divisors(30) =`
      `30 15 10 6 5 3 2 1`

```
void print_divisors (int n)
{
   cout << n << endl;
   int trial_divisor;
   for (trial_divisor = n / 2;
        trial_divisor > = 1;
        --trial_divisor)
      if (/*Enter code here to check
            if n is divisible by
            trial_divisor*/)
   cout << trial_divisor << endl;
   cout << endl;
}

int main()
{
      /*Enter code here*/
}
```

6. What will the following code fragments print to the console? Why? Explain in terms of the trace of input/output and how the function is overloaded.

a.
```
int myfunc(double n) {
    return n * 2.0;
}
int myfunc(float n) {
    return n * 3.0;
}
int main() {
    cout << myfunc(3.5) << endl;
}
```

(a) SOLN: Most compilers will default to the overloaded function that uses a double, so the output will be 7.

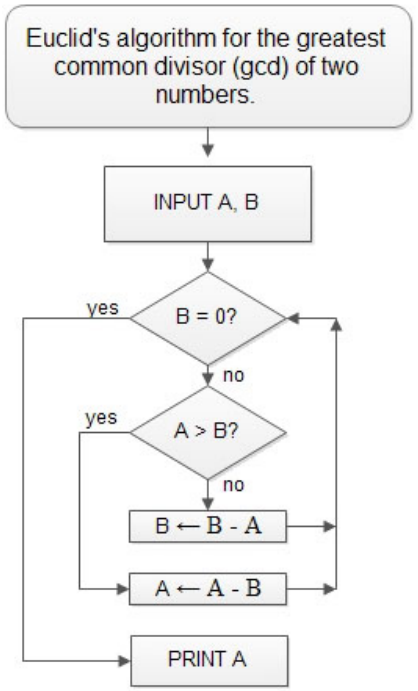
(b) The call to `sum(a1,SIZE)` will sum all zeros. The call to `sum(a2,SIZE)` will sum doubles `0.1+0.4+0.9` so the output will be
0
1.4

b.
```
#define SIZE 4
int sum(int v[], int n) {
   int value = 0;
   for(int i=0; i<n; ++i) {
      value += v[i]/10;   }
   return value;}
double sum(double v[], int n) {
   double value = 0;
    for(int i=0; i<n; ++i) {
       value += v[i]/10.;   }
   return value;}
int main() {
   int a1[SIZE];
   double a2[SIZE];
   for (int i = 0; i < SIZE; i++) {
      a1[i] = i*i;
      a2[i] = i*i;   }
   cout << sum(a1, SIZE) << endl;
   cout << sum(a2, SIZE) << endl;
   return 0; }
```

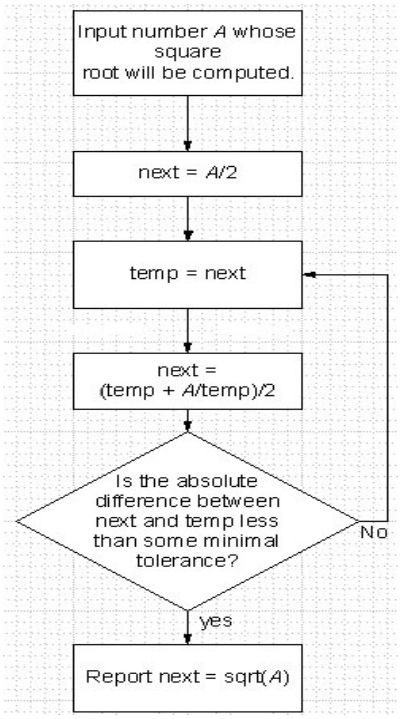
7. How much memory is reserved by each of the following allocation?
   Assume that one char requires 1 byte, an int requires 4 bytes and a double requires 8 bytes.
   a. `unsigned numbs[10];`
      SOLN: `10 ints` requires `40` bytes.
   b. `char word1[10];`
      SOLN: `10 chars` requires `10` bytes.
   c. `double matrix[10][10];`
      SOLN: `100 doubles` requires `800` bytes.

8. Write C++ programs to implement the algorithm described by each flow chart below.

a.



b.



(a) SOLN:

1.
```
#include <iostream>
using namespace std;
int main()  {
      int A, B;
      cout << "\nEnter two positive and the gcd "
           << "\nwill be computed by Euclid: ";
      cin >> A >> B;
      while(B != 0)  {
            if(A > B) A -= B;
            else B -= A;
      }
      cout << "\ngcd = " << A << endl;
}
```

(b) SOLN:

```
#include <iostream>
using namespace std;
double abs(double x) { return x > 0 ? x : -x; }

int main()  {
      double A, temp, next, toler = 1e-10;
      cout << "\nEnter a positive number and this"
            <<  "program will use the "
            << "\nBabylonian algorithm to compute"
            << "the square root: ";
      cin >> A;
      temp = A/2.;
      next = (temp + A/temp)/2.;
      while(abs(temp-next)>toler)   {
            temp = next;
            next = (temp + A/temp)/2.;
      }
      cout << "\nsqrt(" << A << ")=" << next<<endl;
}
```

9. Write a function, `get_number_from_user()` meet the specificationm to meet the specification as described below.

```
/************* GET NUMBER FROM USER ***************
DESCRIPTION: This function prompts an interactive user for a positive
integer, reads the integer, and returns it in the variable passed to this
function.
PARAMETER:
n: The positive integer supplied by the user. This is a reference
parameter. It is intended that the argument passed through this parameter
will receive a new value.
RETURNS: void (no value).
ALGORITHM: The user is prompted for the positive integer. If the integer
entered by the user is zero or negative, the user will be informed that
the input is not positive and will be given another chance to enter valid
data. The function will not exit until the user has entered valid data.
*********************************************************/
SOLN:
void get_number_from_user (int & n)
{
   do
   {
      cout << "Please enter a positive integer: ";
      cin >> n;
      if (n <= 0)
         cout << '\n' << n << " is not a positive integer.\n";
   }
   while (n <= 0);
}
```