CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 1111 0010 from base 2 to hexadecimal f2

   b. (3 pts) Convert 1100 0001 0011 0101 from binary to hexadecimal c135

   c. (3 pts) Convert 101 000 011 from base 2 to base 8 503

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11111111 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -1

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node w;
  int x;
  double y;
  char z;
  Node *a;
  int *b;
  double *c;
  char *d;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argv[1][2] char

b. (3 pts) d char *

c. (3 pts) argc int

d. (3 pts) a->data int

e. (3 pts) &y double *

f. (3 pts) w Node

g. (3 pts) a->next Node *

h. (3 pts) argv[0] char *

i. (3 pts) &d char **

j. (3 pts) *d char

k. (3 pts) a->next->next Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt lemon grape apple
```

a. (2 pts) What is the value of `argc` in this case? 4

b. (2 pts) What is the value of argv[2][3]? p

c. (2 pts) What is the value of argv[1][2]? m

d. (2 pts) What is the value of argv[0][5]? I

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 3254 from hexadecimal to base 2 0011 0010 0101 0100

   b. (3 pts) Convert 92ba from base 16 to base 2 1001 0010 1011 1010

   c. (3 pts) Convert d7f7 from base 16 to base 2 1101 0111 1111 0111

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11100110 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -26

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node x;
  double y;
  int z;
  char a;
  Node *b;
  double *c;
  int *d;
  char *e;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) b->next Node *

b. (3 pts) &b Node **

c. (3 pts) b->data int

d. (3 pts) &x Node *

e. (3 pts) argv[0] char *

f. (3 pts) *e char

g. (3 pts) argc int

h. (3 pts) b->next->next Node *

i. (3 pts) argv[1][2] char

j. (3 pts) d int *

k. (3 pts) z int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt kiwi apple
```

a. (2 pts) What is the value of `argc` in this case? 3

b. (2 pts) What is the value of argv[1][2]? w

c. (2 pts) What is the value of argv[2][4]? e

d. (2 pts) What is the value of argv[0][6]? t

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 35 from base 8 to binary 011 101

   b. (3 pts) Convert 010 011 001 from binary to octal 231

   c. (3 pts) Convert c5d8 from hexadecimal to base 2 1100 0101 1101 1000

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11000100 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -60

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double c;
  int d;
  Node e;
  char f;
  double *g;
  int *h;
  Node *p;
  char *q;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) p->next Node *

b. (3 pts) p->data int

c. (3 pts) c double

d. (3 pts) &f char *

e. (3 pts) *g double

f. (3 pts) argv[1][2] char

g. (3 pts) p->next->next Node *

h. (3 pts) argc int

i. (3 pts) argv[0] char *

j. (3 pts) h int *

k. (3 pts) &q char **

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt banana kiwi apple fig
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[0][3]? u

c. (2 pts) What is the value of argv[1][1]? a

d. (2 pts) What is the value of argv[2][2]? w

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 63 from octal to binary 110 011

   b. (3 pts) Convert 239 from decimal to base 2 1110 1111

   c. (3 pts) Convert 0110 1011 0011 1010 from base 2 to hexadecimal 6b3a

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10011000 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -104

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int t;
  Node w;
  double x;
  char y;
  int *z;
  Node *a;
  double *b;
  char *c;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) a->next->next Node *

b. (3 pts) argv[0] char *

c. (3 pts) a->next Node *

d. (3 pts) &a Node **

e. (3 pts) y char

f. (3 pts) &y char *

g. (3 pts) argc int

h. (3 pts) *z int

i. (3 pts) c char *

j. (3 pts) argv[1][2] char

k. (3 pts) a->data int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt apple cherry mango
```

   a. (2 pts) What is the value of `argc` in this case? 4

   b. (2 pts) What is the value of argv[2][2]? e

   c. (2 pts) What is the value of argv[0][1]? /

   d. (2 pts) What is the value of argv[1][3]? l

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 20 from base 10 to binary 0001 0100

   b. (3 pts) Convert 52 from octal to base 2 101 010

   c. (3 pts) Convert 89 from decimal to binary 0101 1001

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11110101 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -11

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node q;
  int r;
  double s;
  char t;
  Node *w;
  int *x;
  double *y;
  char *z;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) *x int

b. (3 pts) argv[1][2] char

c. (3 pts) w->data int

d. (3 pts) w->next Node *

e. (3 pts) argc int

f. (3 pts) &w Node **

g. (3 pts) &s double *

h. (3 pts) w->next->next Node *

i. (3 pts) s double

j. (3 pts) argv[0] char *

k. (3 pts) y double *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt apple kiwi lemon
```

   a. (2 pts) What is the value of `argc` in this case? 4

   b. (2 pts) What is the value of argv[0][3]? u

   c. (2 pts) What is the value of argv[2][3]? i

   d. (2 pts) What is the value of argv[1][4]? e

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 010 001 011 from binary to base 8 213

   b. (3 pts) Convert 36 from base 8 to base 2 011 110

   c. (3 pts) Convert 100 011 110 from binary to base 8 436

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11011101 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -35

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node r;
  double s;
  int t;
  char w;
  Node *x;
  double *y;
  int *z;
  char *a;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) w char

b. (3 pts) x->data int

c. (3 pts) argv[0] char *

d. (3 pts) argv[1][2] char

e. (3 pts) x->next Node *

f. (3 pts) y double *

g. (3 pts) argc int

h. (3 pts) &y double **

i. (3 pts) x->next->next Node *

j. (3 pts) *y double

k. (3 pts) &s double *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt kiwi lemon
```

   a. (2 pts) What is the value of `argc` in this case? 3

   b. (2 pts) What is the value of argv[1][1]? i

   c. (2 pts) What is the value of argv[2][2]? m

   d. (2 pts) What is the value of argv[0][3]? u

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 8ab0 from hexadecimal to base 2 1000 1010 1011 0000

   b. (3 pts) Convert 0011 0101 from base 2 to decimal 53

   c. (3 pts) Convert 0111 1101 0101 1001 from binary to hexadecimal 7d59

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10111010 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -70

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int h;
  double p;
  Node q;
  char r;
  int *s;
  double *t;
  Node *w;
  char *x;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) *w Node

b. (3 pts) argv[0] char *

c. (3 pts) w->data int

d. (3 pts) &x char **

e. (3 pts) p double

f. (3 pts) s int *

g. (3 pts) w->next Node *

h. (3 pts) argv[1][2] char

i. (3 pts) &h int *

j. (3 pts) w->next->next Node *

k. (3 pts) argc int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt apple date cherry kiwi
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[0][0]? .

c. (2 pts) What is the value of argv[1][0]? a

d. (2 pts) What is the value of argv[2][3]? e

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert e30d from hexadecimal to base 2 1110 0011 0000 1101

   b. (3 pts) Convert 110 110 000 from base 2 to octal 660

   c. (3 pts) Convert 10 from base 8 to base 2 001 000

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10001110 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -114

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int a;
  Node b;
  double c;
  char d;
  int *e;
  Node *f;
  double *g;
  char *h;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) &a int *

b. (3 pts) &g double **

c. (3 pts) argv[1][2] char

d. (3 pts) a int

e. (3 pts) f->data int

f. (3 pts) argc int

g. (3 pts) f->next Node *

h. (3 pts) argv[0] char *

i. (3 pts) *g double

j. (3 pts) h char *

k. (3 pts) f->next->next Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt banana lemon lime
```

a. (2 pts) What is the value of `argc` in this case? 4

b. (2 pts) What is the value of argv[0][3]? u

c. (2 pts) What is the value of argv[2][1]? e

d. (2 pts) What is the value of argv[1][5]? a

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 001 001 111 from binary to octal 117

   b. (3 pts) Convert 1001 0010 from binary to base 10 146

   c. (3 pts) Convert 109b from hexadecimal to base 2 0001 0000 1001 1011

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11101100 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -20

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int g;
  Node h;
  double p;
  char q;
  int *r;
  Node *s;
  double *t;
  char *w;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argv[1][2] char

b. (3 pts) q char

c. (3 pts) &q char *

d. (3 pts) *w char

e. (3 pts) s->data int

f. (3 pts) argc int

g. (3 pts) argv[0] char *

h. (3 pts) s->next Node *

i. (3 pts) s->next->next Node *

j. (3 pts) t double *

k. (3 pts) &r int **

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

   Further, suppose this program is invoked with the following command line:

```
./runIt lemon grape banana
```

   a. (2 pts) What is the value of `argc` in this case? 4

   b. (2 pts) What is the value of argv[2][1]? r

   c. (2 pts) What is the value of argv[0][1]? /

   d. (2 pts) What is the value of argv[1][2]? m

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0101 1001 0100 1110 from base 2 to hexadecimal 594e

   b. (3 pts) Convert 100 from base 10 to binary 0110 0100

   c. (3 pts) Convert 0100 0110 from binary to decimal 70

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11010011 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -45

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double g;
  Node h;
  int p;
  char q;
  double *r;
  Node *s;
  int *t;
  char *w;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) s Node *

b. (3 pts) argv[0] char *

c. (3 pts) &h Node *

d. (3 pts) g double

e. (3 pts) s->data int

f. (3 pts) &t int **

g. (3 pts) s->next->next Node *

h. (3 pts) argc int

i. (3 pts) argv[1][2] char

j. (3 pts) s->next Node *

k. (3 pts) *w char

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt cherry grape
```

a. (2 pts) What is the value of `argc` in this case? 3

b. (2 pts) What is the value of argv[2][1]? r

c. (2 pts) What is the value of argv[1][3]? r

d. (2 pts) What is the value of argv[0][4]? n

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 1001 1110 from base 2 to base 10 158

   b. (3 pts) Convert 1e29 from base 16 to base 2 0001 1110 0010 1001

   c. (3 pts) Convert 0011 0100 from base 2 to decimal 52

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10110000 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -80

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double d;
  int e;
  Node f;
  char g;
  double *h;
  int *p;
  Node *q;
  char *r;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) h double *

b. (3 pts) argc int

c. (3 pts) q->next->next Node *

d. (3 pts) q->next Node *

e. (3 pts) &h double **

f. (3 pts) *h double

g. (3 pts) q->data int

h. (3 pts) &e int *

i. (3 pts) argv[0] char *

j. (3 pts) f Node

k. (3 pts) argv[1][2] char

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt date banana guava lime
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[1][0]? d

c. (2 pts) What is the value of argv[0][2]? r

d. (2 pts) What is the value of argv[2][0]? b

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 010 110 101 from binary to octal 265

   b. (3 pts) Convert 7bdc from base 16 to base 2 0111 1011 1101 1100

   c. (3 pts) Convert 7 from base 8 to base 2 111

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11010010 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -46

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //    appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int r;
  Node s;
  double t;
  char w;
  int *x;
  Node *y;
  double *z;
  char *a;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) y->data int

b. (3 pts) y->next Node *

c. (3 pts) argv[1][2] char

d. (3 pts) y Node *

e. (3 pts) s Node

f. (3 pts) argv[0] char *

g. (3 pts) &y Node **

h. (3 pts) argc int

i. (3 pts) &r int *

j. (3 pts) y->next->next Node *

k. (3 pts) *x int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt banana cherry fig
```

a. (2 pts) What is the value of `argc` in this case? 4

b. (2 pts) What is the value of argv[2][3]? r

c. (2 pts) What is the value of argv[0][3]? u

d. (2 pts) What is the value of argv[1][2]? n

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 9f80 from hexadecimal to base 2 1001 1111 1000 0000

   b. (3 pts) Convert 0011 0110 0000 0101 from binary to base 16 3605

   c. (3 pts) Convert 1100 0001 0001 from binary to base 16 c11

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10110000 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -80

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node h;
  double p;
  int q;
  char r;
  Node *s;
  double *t;
  int *w;
  char *x;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) &r char *

b. (3 pts) argc int

c. (3 pts) s->data int

d. (3 pts) argv[0] char *

e. (3 pts) s->next Node *

f. (3 pts) &s Node **

g. (3 pts) s->next->next Node *

h. (3 pts) s Node *

i. (3 pts) r char

j. (3 pts) *w int

k. (3 pts) argv[1][2] char

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt fig date
```

   a. (2 pts) What is the value of `argc` in this case? 3

   b. (2 pts) What is the value of argv[0][1]? /

   c. (2 pts) What is the value of argv[2][1]? a

   d. (2 pts) What is the value of argv[1][0]? f

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 208 from base 10 to base 2 1101 0000

   b. (3 pts) Convert 001 111 from binary to base 8 17

   c. (3 pts) Convert 0100 0010 from binary to base 10 66

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10010111 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -105

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double h;
  int p;
  Node q;
  char r;
  double *s;
  int *t;
  Node *w;
  char *x;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argv[0] char *

b. (3 pts) w->next Node *

c. (3 pts) &t int **

d. (3 pts) h double

e. (3 pts) &q Node *

f. (3 pts) w->next->next Node *

g. (3 pts) argv[1][2] char

h. (3 pts) *t int

i. (3 pts) argc int

j. (3 pts) s double *

k. (3 pts) w->data int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

   Further, suppose this program is invoked with the following command line:

```
./runIt date mango guava fig
```

   a. (2 pts) What is the value of `argc` in this case? 5

   b. (2 pts) What is the value of argv[1][1]? a

   c. (2 pts) What is the value of argv[2][2]? n

   d. (2 pts) What is the value of argv[0][6]? t

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: _________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0001 0101 1100 0001 from binary to hexadecimal 15c1

   b. (3 pts) Convert 110 000 011 from base 2 to octal 603

   c. (3 pts) Convert 14 from base 8 to base 2 001 100

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11110101 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -11

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int w;
  double x;
  Node y;
  char z;
  int *a;
  double *b;
  Node *c;
  char *d;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argv[1][2] char

b. (3 pts) *d char

c. (3 pts) c->next Node *

d. (3 pts) argv[0] char *

e. (3 pts) argc int

f. (3 pts) c->data int

g. (3 pts) c->next->next Node *

h. (3 pts) y Node

i. (3 pts) &d char **

j. (3 pts) &x double *

k. (3 pts) d char *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt kiwi date mango apple
```

   a. (2 pts) What is the value of `argc` in this case? 5

   b. (2 pts) What is the value of argv[2][1]? a

   c. (2 pts) What is the value of argv[1][0]? k

   d. (2 pts) What is the value of argv[0][3]? u

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0110 1110 0001 1110 from base 2 to hexadecimal 6e1e

   b. (3 pts) Convert 31 from base 8 to binary 011 001

   c. (3 pts) Convert 110 101 011 from binary to octal 653

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11001000 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -56

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node g;
  int h;
  double p;
  char q;
  Node *r;
  int *s;
  double *t;
  char *w;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) p double

b. (3 pts) argv[0] char *

c. (3 pts) *t double

d. (3 pts) &h int *

e. (3 pts) s int *

f. (3 pts) argc int

g. (3 pts) &t double **

h. (3 pts) r->next->next Node *

i. (3 pts) r->data int

j. (3 pts) argv[1][2] char

k. (3 pts) r->next Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt lemon grape cherry
```

   a. (2 pts) What is the value of `argc` in this case? 4

   b. (2 pts) What is the value of argv[0][1]? /

   c. (2 pts) What is the value of argv[2][1]? r

   d. (2 pts) What is the value of argv[1][0]? l

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 1011 0010 from binary to decimal 178

   b. (3 pts) Convert 30 from base 10 to base 2 0001 1110

   c. (3 pts) Convert 110 000 111 from binary to octal 607

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10100110 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -90

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node d;
  double e;
  int f;
  char g;
  Node *h;
  double *p;
  int *q;
  char *r;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) h->data int

b. (3 pts) &h Node **

c. (3 pts) argv[1][2] char

d. (3 pts) &d Node *

e. (3 pts) h Node *

f. (3 pts) h->next->next Node *

g. (3 pts) argc int

h. (3 pts) *h Node

i. (3 pts) argv[0] char *

j. (3 pts) h->next Node *

k. (3 pts) d Node

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt cherry fig
```

a. (2 pts) What is the value of `argc` in this case? 3

b. (2 pts) What is the value of argv[2][0]? f

c. (2 pts) What is the value of argv[0][3]? u

d. (2 pts) What is the value of argv[1][5]? y

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 71 from base 8 to base 2 111 001

   b. (3 pts) Convert 1111 0000 from binary to decimal 240

   c. (3 pts) Convert 249 from decimal to base 2 1111 1001

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10001101 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -115

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double e;
  int f;
  Node g;
  char h;
  double *p;
  int *q;
  Node *r;
  char *s;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) r->next->next Node *

b. (3 pts) r->data int

c. (3 pts) f int

d. (3 pts) s char *

e. (3 pts) argv[1][2] char

f. (3 pts) *s char

g. (3 pts) &r Node **

h. (3 pts) argv[0] char *

i. (3 pts) r->next Node *

j. (3 pts) argc int

k. (3 pts) &g Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt guava apple fig date
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[1][4]? a

c. (2 pts) What is the value of argv[2][3]? l

d. (2 pts) What is the value of argv[0][2]? r

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ____________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 41 from base 10 to base 2 0010 1001

   b. (3 pts) Convert 170 from base 10 to base 2 1010 1010

   c. (3 pts) Convert 231 from base 10 to binary 1110 0111

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11101011 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -21

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double a;
  int b;
  Node c;
  char d;
  double *e;
  int *f;
  Node *g;
  char *h;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argv[0] char *

b. (3 pts) *f int

c. (3 pts) g Node *

d. (3 pts) &b int *

e. (3 pts) g->data int

f. (3 pts) g->next->next Node *

g. (3 pts) d char

h. (3 pts) &e double **

i. (3 pts) g->next Node *

j. (3 pts) argv[1][2] char

k. (3 pts) argc int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt mango cherry banana guava
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[0][4]? n

c. (2 pts) What is the value of argv[1][3]? g

d. (2 pts) What is the value of argv[2][3]? r

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 129 from base 10 to base 2 1000 0001

   b. (3 pts) Convert 010 011 010 from base 2 to octal 232

   c. (3 pts) Convert 8d33 from hexadecimal to base 2 1000 1101 0011 0011

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10111111 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -65

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node r;
  int s;
  double t;
  char w;
  Node *x;
  int *y;
  double *z;
  char *a;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) w   char

b. (3 pts) x->next   Node *

c. (3 pts) argc   int

d. (3 pts) &a   char **

e. (3 pts) *x   Node

f. (3 pts) argv[0]   char *

g. (3 pts) argv[1][2]   char

h. (3 pts) &s   int *

i. (3 pts) x->data   int

j. (3 pts) y   int *

k. (3 pts) x->next->next   Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt date apple guava
```

   a. (2 pts) What is the value of `argc` in this case? 4

   b. (2 pts) What is the value of argv[0][6]? t

   c. (2 pts) What is the value of argv[1][1]? a

   d. (2 pts) What is the value of argv[2][3]? l

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 61 from base 8 to binary 110 001

   b. (3 pts) Convert 74b from base 16 to base 2 0111 0100 1011

   c. (3 pts) Convert 0111 1011 from base 2 to base 10 123

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10011100 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -100

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double z;
  Node a;
  int b;
  char c;
  double *d;
  Node *e;
  int *f;
  char *g;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) e->next Node *

b. (3 pts) d double *

c. (3 pts) argc int

d. (3 pts) e->data int

e. (3 pts) e->next->next Node *

f. (3 pts) argv[1][2] char

g. (3 pts) *f int

h. (3 pts) argv[0] char *

i. (3 pts) a Node

j. (3 pts) &e Node **

k. (3 pts) &z double *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

   Further, suppose this program is invoked with the following command line:

```
./runIt date fig
```

   a. (2 pts) What is the value of `argc` in this case? 3

   b. (2 pts) What is the value of argv[1][3]? e

   c. (2 pts) What is the value of argv[2][0]? f

   d. (2 pts) What is the value of argv[0][4]? n

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert f7dc from hexadecimal to binary 1111 0111 1101 1100

   b. (3 pts) Convert d8d1 from hexadecimal to binary 1101 1000 1101 0001

   c. (3 pts) Convert 54 from base 8 to binary 101 100

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10000100 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -124

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int a;
  double b;
  Node c;
  char d;
  int *e;
  double *f;
  Node *g;
  char *h;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) g->data int

b. (3 pts) &g Node **

c. (3 pts) g->next Node *

d. (3 pts) argv[0] char *

e. (3 pts) argc int

f. (3 pts) argv[1][2] char

g. (3 pts) &d char *

h. (3 pts) h char *

i. (3 pts) g->next->next Node *

j. (3 pts) c Node

k. (3 pts) *f double

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt fig grape lime lemon
```

   a. (2 pts) What is the value of `argc` in this case? 5

   b. (2 pts) What is the value of argv[2][1]? r

   c. (2 pts) What is the value of argv[1][2]? g

   d. (2 pts) What is the value of argv[0][6]? t

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ____________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 001 111 001 from binary to base 8 171

   b. (3 pts) Convert 44 from octal to base 2 100 100

   c. (3 pts) Convert 9f53 from hexadecimal to binary 1001 1111 0101 0011

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11100001 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -31

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int x;
  double y;
  Node z;
  char a;
  int *b;
  double *c;
  Node *d;
  char *e;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) y   double

b. (3 pts) argv[0]   char *

c. (3 pts) &z   Node *

d. (3 pts) d->next   Node *

e. (3 pts) *e   char

f. (3 pts) d   Node *

g. (3 pts) d->next->next   Node *

h. (3 pts) argc   int

i. (3 pts) &c   double **

j. (3 pts) d->data   int

k. (3 pts) argv[1][2]   char

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt kiwi grape guava cherry
```

   a. (2 pts) What is the value of `argc` in this case? 5

   b. (2 pts) What is the value of argv[1][0]? k

   c. (2 pts) What is the value of argv[0][2]? r

   d. (2 pts) What is the value of argv[2][0]? g

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 100 101 010 from binary to base 8 452

   b. (3 pts) Convert 0011 0101 1100 0110 from binary to hexadecimal 35c6

   c. (3 pts) Convert 0100 0100 1011 0101 from base 2 to base 16 44b5

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10110101 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -75

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int h;
  Node p;
  double q;
  char r;
  int *s;
  Node *t;
  double *w;
  char *x;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) &x char **

b. (3 pts) argv[0] char *

c. (3 pts) &q double *

d. (3 pts) argv[1][2] char

e. (3 pts) t->next Node *

f. (3 pts) h int

g. (3 pts) t->next->next Node *

h. (3 pts) *w double

i. (3 pts) t->data int

j. (3 pts) argc int

k. (3 pts) s int *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt fig lime apple
```

   a. (2 pts) What is the value of `argc` in this case? 4

   b. (2 pts) What is the value of argv[0][5]? I

   c. (2 pts) What is the value of argv[2][2]? m

   d. (2 pts) What is the value of argv[1][0]? f

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: _________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 217 from base 10 to base 2 1101 1001

   b. (3 pts) Convert 73 from base 8 to binary 111 011

   c. (3 pts) Convert 0011 0010 1001 0101 from binary to hexadecimal 3295

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 10010011 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -109

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double e;
  Node f;
  int g;
  char h;
  double *p;
  Node *q;
  int *r;
  char *s;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) &r int **

b. (3 pts) q->next Node *

c. (3 pts) &f Node *

d. (3 pts) argv[0] char *

e. (3 pts) g int

f. (3 pts) argv[1][2] char

g. (3 pts) q->next->next Node *

h. (3 pts) s char *

i. (3 pts) q->data int

j. (3 pts) argc int

k. (3 pts) *p double

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt date grape
```

   a. (2 pts) What is the value of `argc` in this case? 3

   b. (2 pts) What is the value of argv[2][0]? g

   c. (2 pts) What is the value of argv[0][4]? n

   d. (2 pts) What is the value of argv[1][0]? d

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 1011 from base 2 to decimal 11

   b. (3 pts) Convert 1100 0001 0111 0100 from binary to hexadecimal c174

   c. (3 pts) Convert 32 from octal to base 2 011 010

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11111010 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -6

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double w;
  Node x;
  int y;
  char z;
  double *a;
  Node *b;
  int *c;
  char *d;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argv[1][2] char

b. (3 pts) *d char

c. (3 pts) d char *

d. (3 pts) z char

e. (3 pts) argv[0] char *

f. (3 pts) b->next->next Node *

g. (3 pts) b->data int

h. (3 pts) &d char **

i. (3 pts) b->next Node *

j. (3 pts) &z char *

k. (3 pts) argc int

6. Assume the `main` function in the program `runIt.cpp` starts with:

   ```
   int main(int argc, char *argv[]) {
   ...
   ```

   Further, suppose this program is invoked with the following command line:

   ```
   ./runIt date lime
   ```

   a. (2 pts) What is the value of `argc` in this case? 3

   b. (2 pts) What is the value of argv[2][3]? e

   c. (2 pts) What is the value of argv[1][1]? a

   d. (2 pts) What is the value of argv[0][0]? .

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 5039 from hexadecimal to binary 0101 0000 0011 1001

   b. (3 pts) Convert 0111 1011 from binary to base 10 123

   c. (3 pts) Convert 010 101 101 from binary to base 8 255

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -93, what is this number's binary representation in 8-bit two's complement?

      10100011

   b. (3 pts) Given that 11010111 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -41

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int r;
  double s;
  Node t;
  char w;
  int *x;
  double *y;
  Node *z;
  char *a;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) &y double **

b. (3 pts) y double *

c. (3 pts) &t Node *

d. (3 pts) z->next Node *

e. (3 pts) argc int

f. (3 pts) argv[1][2] char

g. (3 pts) z->next->next Node *

h. (3 pts) t Node

i. (3 pts) argv[0] char *

j. (3 pts) *y double

k. (3 pts) z->data int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt guava lime cherry grape
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[1][2]? a

c. (2 pts) What is the value of argv[2][3]? e

d. (2 pts) What is the value of argv[0][4]? n

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0110 0101 from binary to base 10 101

   b. (3 pts) Convert 3ab9 from base 16 to binary 0011 1010 1011 1001

   c. (3 pts) Convert 1101 0111 1110 0010 from base 2 to base 16 d7e2

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 11001101 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -51

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int g;
  Node h;
  double p;
  char q;
  int *r;
  Node *s;
  double *t;
  char *w;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argv[1][2] char

b. (3 pts) &w char **

c. (3 pts) s->data int

d. (3 pts) s->next Node *

e. (3 pts) *r int

f. (3 pts) r int *

g. (3 pts) argv[0] char *

h. (3 pts) &h Node *

i. (3 pts) p double

j. (3 pts) argc int

k. (3 pts) s->next->next Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt kiwi cherry fig
```

a. (2 pts) What is the value of `argc` in this case? 4

b. (2 pts) What is the value of argv[2][1]? h

c. (2 pts) What is the value of argv[1][3]? i

d. (2 pts) What is the value of argv[0][5]? I

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert aa12 from hexadecimal to binary 1010 1010 0001 0010

   b. (3 pts) Convert 111 101 001 from base 2 to base 8 751

   c. (3 pts) Convert 197 from decimal to binary 1100 0101

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 10101010 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -86

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node s;
  double t;
  int w;
  char x;
  Node *y;
  double *z;
  int *a;
  char *b;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argc int

b. (3 pts) &z double **

c. (3 pts) argv[0] char *

d. (3 pts) s Node

e. (3 pts) y->data int

f. (3 pts) y->next->next Node *

g. (3 pts) y->next Node *

h. (3 pts) argv[1][2] char

i. (3 pts) b char *

j. (3 pts) *a int

k. (3 pts) &s Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt guava banana
```

a. (2 pts) What is the value of `argc` in this case? 3

b. (2 pts) What is the value of argv[2][2]? n

c. (2 pts) What is the value of argv[0][0]? .

d. (2 pts) What is the value of argv[1][1]? u

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 219 from base 10 to base 2 1101 1011

   b. (3 pts) Convert 61 from octal to binary 110 001

   c. (3 pts) Convert 77 from octal to base 2 111 111

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 10010010 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -110

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double t;
  int w;
  Node x;
  char y;
  double *z;
  int *a;
  Node *b;
  char *c;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) b->next->next   Node *

b. (3 pts) &y   char *

c. (3 pts) b->next   Node *

d. (3 pts) argc   int

e. (3 pts) argv[1][2]   char

f. (3 pts) w   int

g. (3 pts) argv[0]   char *

h. (3 pts) b->data   int

i. (3 pts) &b   Node **

j. (3 pts) c   char *

k. (3 pts) *a   int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt banana apple cherry mango
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[0][5]? I

c. (2 pts) What is the value of argv[2][1]? p

d. (2 pts) What is the value of argv[1][4]? n

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0010 0000 0101 0100 from base 2 to base 16 2054

   b. (3 pts) Convert 128 from decimal to base 2 1000 0000

   c. (3 pts) Convert ea01 from hexadecimal to base 2 1110 1010 0000 0001

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 11101111 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -17

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double q;
  int r;
  Node s;
  char t;
  double *w;
  int *x;
  Node *y;
  char *z;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) y->next->next Node *

b. (3 pts) y->data int

c. (3 pts) *z char

d. (3 pts) argc int

e. (3 pts) y Node *

f. (3 pts) q double

g. (3 pts) y->next Node *

h. (3 pts) argv[0] char *

i. (3 pts) &s Node *

j. (3 pts) &w double **

k. (3 pts) argv[1][2] char

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt mango apple banana date
```

   a. (2 pts) What is the value of `argc` in this case? 5

   b. (2 pts) What is the value of argv[1][0]? m

   c. (2 pts) What is the value of argv[2][1]? p

   d. (2 pts) What is the value of argv[0][1]? /

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 36 from base 8 to base 2 011 110

   b. (3 pts) Convert 10 from octal to binary 001 000

   c. (3 pts) Convert 1000 1111 0110 0011 from binary to hexadecimal 8f63

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 11000011 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -61

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node c;
  int d;
  double e;
  char f;
  Node *g;
  int *h;
  double *p;
  char *q;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) *p double

b. (3 pts) f char

c. (3 pts) g->next Node *

d. (3 pts) g->next->next Node *

e. (3 pts) &e double *

f. (3 pts) &q char **

g. (3 pts) g->data int

h. (3 pts) argc int

i. (3 pts) argv[0] char *

j. (3 pts) argv[1][2] char

k. (3 pts) g Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

   Further, suppose this program is invoked with the following command line:

```
./runIt apple lemon lime
```

   a. (2 pts) What is the value of `argc` in this case? 4

   b. (2 pts) What is the value of argv[0][4]? n

   c. (2 pts) What is the value of argv[1][2]? p

   d. (2 pts) What is the value of argv[2][1]? e

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 1011 1101 from base 2 to base 10 189

   b. (3 pts) Convert 221 from base 10 to base 2 1101 1101

   c. (3 pts) Convert 37 from octal to binary 011 111

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 11001110 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -50

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double z;
  Node a;
  int b;
  char c;
  double *d;
  Node *e;
  int *f;
  char *g;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) &e Node **

b. (3 pts) argc int

c. (3 pts) argv[0] char *

d. (3 pts) &a Node *

e. (3 pts) a Node

f. (3 pts) e->next Node *

g. (3 pts) argv[1][2] char

h. (3 pts) g char *

i. (3 pts) e->next->next Node *

j. (3 pts) e->data int

k. (3 pts) *d double

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt lime fig
```

a. (2 pts) What is the value of `argc` in this case? 3

b. (2 pts) What is the value of argv[1][2]? m

c. (2 pts) What is the value of argv[0][5]? I

d. (2 pts) What is the value of argv[2][1]? i

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: _______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 111 011 101 from binary to octal 735

   b. (3 pts) Convert 1010 1111 from binary to decimal 175

   c. (3 pts) Convert 101 100 111 from binary to octal 547

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 10001000 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -120

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int p;
  double q;
  Node r;
  char s;
  int *t;
  double *w;
  Node *x;
  char *y;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argv[1][2] char

b. (3 pts) s char

c. (3 pts) argc int

d. (3 pts) x Node *

e. (3 pts) *y char

f. (3 pts) &s char *

g. (3 pts) x->data int

h. (3 pts) &y char **

i. (3 pts) x->next Node *

j. (3 pts) x->next->next Node *

k. (3 pts) argv[0] char *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt lime lemon guava cherry
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[0][2]? r

c. (2 pts) What is the value of argv[2][4]? n

d. (2 pts) What is the value of argv[1][1]? i

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 51 from base 10 to base 2 0011 0011

   b. (3 pts) Convert 0110 1001 0011 0011 from binary to base 16 6933

   c. (3 pts) Convert 101 000 011 from base 2 to octal 503

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 11100110 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -26

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double f;
  int g;
  Node h;
  char p;
  double *q;
  int *r;
  Node *s;
  char *t;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argc   int

b. (3 pts) s->next   Node *

c. (3 pts) *r   int

d. (3 pts) argv[0]   char *

e. (3 pts) &p   char *

f. (3 pts) &r   int **

g. (3 pts) r   int *

h. (3 pts) g   int

i. (3 pts) s->next->next   Node *

j. (3 pts) argv[1][2]   char

k. (3 pts) s->data   int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt grape lemon lime fig
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[2][2]? m

c. (2 pts) What is the value of argv[0][6]? t

d. (2 pts) What is the value of argv[1][3]? p

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 140 from decimal to base 2 1000 1100

   b. (3 pts) Convert 11 from decimal to base 2 1011

   c. (3 pts) Convert 0100 0110 from binary to base 10 70

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 10111001 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -71

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int y;
  Node z;
  double a;
  char b;
  int *c;
  Node *d;
  double *e;
  char *f;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) c int *

b. (3 pts) argc int

c. (3 pts) *c int

d. (3 pts) d->data int

e. (3 pts) argv[0] char *

f. (3 pts) d->next Node *

g. (3 pts) d->next->next Node *

h. (3 pts) argv[1][2] char

i. (3 pts) y int

j. (3 pts) &a double *

k. (3 pts) &c int **

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt grape fig date
```

a. (2 pts) What is the value of `argc` in this case? 4

b. (2 pts) What is the value of argv[1][0]? g

c. (2 pts) What is the value of argv[0][6]? t

d. (2 pts) What is the value of argv[2][2]? g

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 64 from base 8 to base 2 110 100

   b. (3 pts) Convert c628 from base 16 to binary 1100 0110 0010 1000

   c. (3 pts) Convert 15 from base 8 to base 2 001 101

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 10010111 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -105

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double t;
  Node w;
  int x;
  char y;
  double *z;
  Node *a;
  int *b;
  char *c;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) argc int

b. (3 pts) a->next->next Node *

c. (3 pts) &w Node *

d. (3 pts) argv[1][2] char

e. (3 pts) a->data int

f. (3 pts) c char *

g. (3 pts) y char

h. (3 pts) *b int

i. (3 pts) a->next Node *

j. (3 pts) &b int **

k. (3 pts) argv[0] char *

---

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt banana lime
```

   a. (2 pts) What is the value of `argc` in this case? 3

   b. (2 pts) What is the value of argv[0][0]? .

   c. (2 pts) What is the value of argv[1][2]? n

   d. (2 pts) What is the value of argv[2][3]? e

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0010 0110 1111 from binary to hexadecimal 26f

   b. (3 pts) Convert 1001 0111 1010 1110 from base 2 to base 16 97ae

   c. (3 pts) Convert 6b24 from base 16 to base 2 0110 1011 0010 0100

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 11111110 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -2

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node e;
  double f;
  int g;
  char h;
  Node *p;
  double *q;
  int *r;
  char *s;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) r int *

b. (3 pts) *q double

c. (3 pts) p->next Node *

d. (3 pts) &e Node *

e. (3 pts) p->data int

f. (3 pts) p->next->next Node *

g. (3 pts) argc int

h. (3 pts) argv[0] char *

i. (3 pts) &p Node **

j. (3 pts) e Node

k. (3 pts) argv[1][2] char

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt lime apple
```

   a. (2 pts) What is the value of `argc` in this case? 3

   b. (2 pts) What is the value of argv[0][0]? .

   c. (2 pts) What is the value of argv[2][0]? a

   d. (2 pts) What is the value of argv[1][2]? m

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 010 001 110 from binary to octal 216

   b. (3 pts) Convert 24 from octal to binary 010 100

   c. (3 pts) Convert 0101 1001 from binary to base 10 89

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 11011100 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -36

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int b;
  double c;
  Node d;
  char e;
  int *f;
  double *g;
  Node *h;
  char *p;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) h->next Node *

b. (3 pts) argv[0] char *

c. (3 pts) *p char

d. (3 pts) &e char *

e. (3 pts) argv[1][2] char

f. (3 pts) argc int

g. (3 pts) h->data int

h. (3 pts) &h Node **

i. (3 pts) g double *

j. (3 pts) h->next->next Node *

k. (3 pts) d Node

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt fig apple kiwi lime
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[1][2]? g

c. (2 pts) What is the value of argv[2][3]? l

d. (2 pts) What is the value of argv[0][0]? .

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 100 111 111 from binary to octal 477

   b. (3 pts) Convert 1111 0100 1010 0010 from binary to base 16 f4a2

   c. (3 pts) Convert 254 from base 10 to binary 1111 1110

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 10110000 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -80

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.  Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int s;
  Node t;
  double w;
  char x;
  int *y;
  Node *z;
  double *a;
  char *b;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) b char *

b. (3 pts) z->data int

c. (3 pts) t Node

d. (3 pts) z->next Node *

e. (3 pts) *a double

f. (3 pts) &z Node **

g. (3 pts) z->next->next Node *

h. (3 pts) &x char *

i. (3 pts) argc int

j. (3 pts) argv[0] char *

k. (3 pts) argv[1][2] char

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt guava fig apple
```

   a. (2 pts) What is the value of `argc` in this case? 4

   b. (2 pts) What is the value of argv[1][2]? a

   c. (2 pts) What is the value of argv[2][0]? f

   d. (2 pts) What is the value of argv[0][1]? /

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert e48a from hexadecimal to base 2 1110 0100 1000 1010

   b. (3 pts) Convert 1010 1110 from binary to decimal 174

   c. (3 pts) Convert 236 from decimal to binary 1110 1100

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 10001101 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -115

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  Node p;
  double q;
  int r;
  char s;
  Node *t;
  double *w;
  int *x;
  char *y;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) x int *

b. (3 pts) &r int *

c. (3 pts) argc int

d. (3 pts) p Node

e. (3 pts) t->next Node *

f. (3 pts) &y char **

g. (3 pts) argv[1][2] char

h. (3 pts) t->data int

i. (3 pts) *t Node

j. (3 pts) argv[0] char *

k. (3 pts) t->next->next Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt guava cherry
```

a. (2 pts) What is the value of `argc` in this case? 3

b. (2 pts) What is the value of argv[0][4]? n

c. (2 pts) What is the value of argv[2][2]? e

d. (2 pts) What is the value of argv[1][3]? v

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0001 0101 from base 2 to base 10 21

   b. (3 pts) Convert 100 000 000 from base 2 to base 8 400

   c. (3 pts) Convert 0010 0010 1010 0101 from binary to base 16 22a5

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 11110101 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -11

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double q;
  Node r;
  int s;
  char t;
  double *w;
  Node *x;
  int *y;
  char *z;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) *z char

b. (3 pts) x->next Node *

c. (3 pts) argv[1][2] char

d. (3 pts) x->next->next Node *

e. (3 pts) argc int

f. (3 pts) y int *

g. (3 pts) &q double *

h. (3 pts) r Node

i. (3 pts) argv[0] char *

j. (3 pts) &w double **

k. (3 pts) x->data int

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt banana cherry
```

   a. (2 pts) What is the value of `argc` in this case? 3

   b. (2 pts) What is the value of argv[2][1]? h

   c. (2 pts) What is the value of argv[0][5]? I

   d. (2 pts) What is the value of argv[1][2]? n

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0101 1010 1100 1011 from base 2 to hexadecimal 5acb

   b. (3 pts) Convert 0011 1010 from base 2 to decimal 58

   c. (3 pts) Convert 0001 0000 1000 0110 from binary to hexadecimal 1086

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -95, what is this number's binary representation in 8-bit two's complement?

      10100001

   b. (3 pts) Given that 11010010 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -46

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double y;
  int z;
  Node a;
  char b;
  double *c;
  int *d;
  Node *e;
  char *f;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) e->data int

b. (3 pts) e->next Node *

c. (3 pts) argv[0] char *

d. (3 pts) &e Node **

e. (3 pts) c double *

f. (3 pts) e->next->next Node *

g. (3 pts) b char

h. (3 pts) argc int

i. (3 pts) argv[1][2] char

j. (3 pts) *d int

k. (3 pts) &y double *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt mango guava grape apple
```

   a. (2 pts) What is the value of `argc` in this case? 5

   b. (2 pts) What is the value of argv[2][0]? g

   c. (2 pts) What is the value of argv[1][1]? a

   d. (2 pts) What is the value of argv[0][3]? u

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0001 0111 0011 1110 from binary to base 16 173e

   b. (3 pts) Convert 1001 1000 from binary to base 10 152

   c. (3 pts) Convert 249 from base 10 to binary 1111 1001

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 11110100 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -12

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int w;
  double x;
  Node y;
  char z;
  int *a;
  double *b;
  Node *c;
  char *d;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) *b double

b. (3 pts) argc int

c. (3 pts) &z char *

d. (3 pts) argv[1][2] char

e. (3 pts) argv[0] char *

f. (3 pts) c->next->next Node *

g. (3 pts) c->next Node *

h. (3 pts) y Node

i. (3 pts) &a int **

j. (3 pts) c->data int

k. (3 pts) c Node *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt kiwi lime grape date
```

a. (2 pts) What is the value of `argc` in this case? 5

b. (2 pts) What is the value of argv[2][1]? i

c. (2 pts) What is the value of argv[1][2]? w

d. (2 pts) What is the value of argv[0][1]? /

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 010 111 000 from binary to base 8 270

   b. (3 pts) Convert 0101 0010 0101 0101 from binary to hexadecimal 5255

   c. (3 pts) Convert 1110 0111 from base 2 to decimal 231

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 11010001 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -47

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int q;
  Node r;
  double s;
  char t;
  int *w;
  Node *x;
  double *y;
  char *z;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) x->next->next Node *

b. (3 pts) argv[1][2] char

c. (3 pts) &s double *

d. (3 pts) x->data int

e. (3 pts) &y double **

f. (3 pts) x->next Node *

g. (3 pts) argv[0] char *

h. (3 pts) argc int

i. (3 pts) x Node *

j. (3 pts) q int

k. (3 pts) *y double

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt apple kiwi fig
```

a. (2 pts) What is the value of `argc` in this case? 4

b. (2 pts) What is the value of argv[1][4]? e

c. (2 pts) What is the value of argv[0][3]? u

d. (2 pts) What is the value of argv[2][0]? k

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 43 from base 8 to binary 100 011

   b. (3 pts) Convert 23db from base 16 to base 2 0010 0011 1101 1011

   c. (3 pts) Convert 0001 1110 0001 1011 from base 2 to hexadecimal 1e1b

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 10111001 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -71

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double c;
  Node d;
  int e;
  char f;
  double *g;
  Node *h;
  int *p;
  char *q;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) h->data int

b. (3 pts) &d Node *

c. (3 pts) g double *

d. (3 pts) h->next->next Node *

e. (3 pts) d Node

f. (3 pts) argc int

g. (3 pts) *p int

h. (3 pts) argv[1][2] char

i. (3 pts) argv[0] char *

j. (3 pts) h->next Node *

k. (3 pts) &g double **

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

   Further, suppose this program is invoked with the following command line:

```
./runIt date lime
```

   a. (2 pts) What is the value of `argc` in this case? 3

   b. (2 pts) What is the value of argv[0][5]? I

   c. (2 pts) What is the value of argv[2][2]? m

   d. (2 pts) What is the value of argv[1][3]? e

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ____________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
  - Each exam is numbered (e.g. Exam #137).
  - Each pages is numbered (e.g. Page 1, Page 2, etc.)
  - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 1101 0010 from binary to base 10 210

   b. (3 pts) Convert 1101 1110 0000 0100 from base 2 to hexadecimal de04

   c. (3 pts) Convert 11 from decimal to binary 1011

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 10010110 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -106

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int z;
  double a;
  Node b;
  char c;
  int *d;
  double *e;
  Node *f;
  char *g;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) &f Node **

b. (3 pts) &z int *

c. (3 pts) g char *

d. (3 pts) f->data int

e. (3 pts) argc int

f. (3 pts) c char

g. (3 pts) f->next->next Node *

h. (3 pts) f->next Node *

i. (3 pts) *g char

j. (3 pts) argv[1][2] char

k. (3 pts) argv[0] char *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt date lime grape fig
```

   a. (2 pts) What is the value of `argc` in this case? 5

   b. (2 pts) What is the value of argv[1][2]? t

   c. (2 pts) What is the value of argv[2][0]? l

   d. (2 pts) What is the value of argv[0][3]? u

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0010 1010 from binary to base 10 42

   b. (3 pts) Convert 80d0 from base 16 to base 2 1000 0000 1101 0000

   c. (3 pts) Convert b15d from base 16 to base 2 1011 0001 0101 1101

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 11101010 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -22

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int q;
  double r;
  Node s;
  char t;
  int *w;
  double *x;
  Node *y;
  char *z;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) t char

b. (3 pts) argc int

c. (3 pts) y->next->next Node *

d. (3 pts) *z char

e. (3 pts) y->next Node *

f. (3 pts) argv[1][2] char

g. (3 pts) y Node *

h. (3 pts) y->data int

i. (3 pts) argv[0] char *

j. (3 pts) &x double **

k. (3 pts) &q int *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt lemon apple kiwi cherry
```

   a. (2 pts) What is the value of `argc` in this case? 5

   b. (2 pts) What is the value of argv[1][4]? n

   c. (2 pts) What is the value of argv[2][1]? p

   d. (2 pts) What is the value of argv[0][4]? n

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: _________________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert 0110 1111 1001 1011 from binary to base 16 6f9b

   b. (3 pts) Convert 001 110 101 from binary to octal 165

   c. (3 pts) Convert 9f3d from hexadecimal to base 2 1001 1111 0011 1101

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 11001000 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -56

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  int g;
  Node h;
  double p;
  char q;
  int *r;
  Node *s;
  double *t;
  char *w;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) h Node

b. (3 pts) s->data int

c. (3 pts) &q char *

d. (3 pts) r int *

e. (3 pts) s->next->next Node *

f. (3 pts) *r int

g. (3 pts) s->next Node *

h. (3 pts) argv[1][2] char

i. (3 pts) argc int

j. (3 pts) &w char **

k. (3 pts) argv[0] char *

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt kiwi mango fig
```

   a. (2 pts) What is the value of `argc` in this case? 4

   b. (2 pts) What is the value of argv[0][3]? u

   c. (2 pts) What is the value of argv[2][4]? o

   d. (2 pts) What is the value of argv[1][1]? i

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

   You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100

CS16—Final Exam

E03, W15, Phill Conrad, UC Santa Barbara

Wednesday, 03/17/2015, 7pm-10pm

Name: ____

Umail Address: ______________________________@ umail.ucsb.edu

- Please write your name **above AND AT THE TOP OF EVERY PAGE**
- Be sure you turn in every page of this exam.
    - Each exam is numbered (e.g. Exam #137).
    - Each pages is numbered (e.g. Page 1, Page 2, etc.)
    - The last page clearly says "End of Exam".
- This exam is **closed book, closed notes**, **closed mouth, cell phone off**
- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

1. a. (3 pts) Convert a0fd from base 16 to binary 1010 0000 1111 1101

   b. (3 pts) Convert 3 from base 8 to base 2 011

   c. (3 pts) Convert 1101 0101 from binary to base 10 213

2. For these questions, assume 8-bit two's complement representation of negative integers. The left-most bit is the sign bit, with 1 representing negative numbers, and 0 representing positive numbers.

   a. (3 pts) Given the decimal number -96, what is this number's binary representation in 8-bit two's complement?

      10100000

   b. (3 pts) Given that 10101111 is the 8-bit two's complement representation of a number, what is that number in base ten?

      -81

3. Fill in the incomplete parts of the function definitions below and on the next page. These functions use structs defined in `recipe.h` which is on a separate handout given with this exam.

   You should find parts (a) and (b) on this page.

   Parts (c), (d), (e) and (f) are on the next page, but this is ALL ONE PROGRAM, and ALL ONE QUESTION.

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

#include "recipe.h"

double totalCalories(Recipe r) {
  double result = 0.0;

  // (a) (4 pts) Add a line of code here that declares the variable n with an
  //   appropriate type for how it is used in the for loop below.



  for (n=r.head; n!=NULL; n=n->next) {
    // (b) (4 pts) Add a line of code here that accumulates the total calories
    //  in the recipe.   Note that for each item in the linked list,
    //  the calories for that item is the result of multiplying
    //  cups times calsPerCup, so it is that product that must be added
    //  to the total.



  }
  return result;
}
```

CONTINUED ON NEXT PAGE

CONTINUED FROM PREVIOUS PAGE

```
// Add Ingredient to a Recipe
void addToRecipe(Recipe *r, Ingredient i) {

  // Declare a variable n of type pointer to IngrNode, and
  // initialize it to a new IngrNode struct on the heap (c) (4 pts)



  n->ing = i;
  n->next = NULL;

  // FILL IN the boolean condition for the if test? (d) (4 pts)

  if (                                        ) {
    r -> head = n;
    r -> tail = n;
  } else {
    r -> tail -> next = n;

    // FILL IN A LINE OF CODE HERE  (e) (4 pts)



  }
}

int main() {
  Ingredient ingredients[] = {
    {"dry steel cut oats", 0.67, 600},
    {"soy milk", 1.67, 110},
    {"maple syrup", 0.125, 840},
    {"vanilla extract", 0.02, 599}, // 1 tsp
    {"cinnamon", 0.02, 288}, // 1 tsp
    {"chopped dates", 0.5, 532},
  };
  Recipe creamyOats = { NULL, NULL};

  for (int i=0; i<6; i++ ) {

    // (f) (4 pts) Fill in a single line of code here that calls the function
    //  addToRecipe each time through the loop to that each element
    //  of ingredients gets added to the Recipe creamyOats.



  }
  double calories  = totalCalories(creamyOats);
  cout << "Total Calories=" << calories << endl;
  return 0;
}
```

4. Most of the code we wrote in this course goes in files that end in .cpp, for example: `arrayFuncs.cpp`, or `arrayFuncTests.cpp`.

   But sometimes we worked with each of the following types of files. For each type, explain what its purpose is, and its relationship to .cpp files.

   DON'T USE TOO MANY WORDS. Write just enough to get the main point across. Excessively verbose answers that stray off point may be penalized.

   a. (2 pts) Purpose of .h files:

   b. (2 pts) .h files relationship to .cpp files:

   c. (2 pts) Purpose of .o files:

   d. (2 pts) .o files relationship to .cpp files.

   e. (2 pts) Purpose of Makefile:

   f. (2 pts) Relationship of Makefile to .cpp files:

5. Given the following declarations:

```
struct Node {
  int data;
  Node *next;
};


int main(int argc, char *argv[]) {
  double y;
  Node z;
  int a;
  char b;
  double *c;
  Node *d;
  int *e;
  char *f;


  return 0;
}
```

Specify the type of each of these expressions (e.g. `int`, `int *`, etc.

a. (3 pts) &c double **

b. (3 pts) argc int

c. (3 pts) *c double

d. (3 pts) d->next Node *

e. (3 pts) argv[0] char *

f. (3 pts) &z Node *

g. (3 pts) d->next->next Node *

h. (3 pts) c double *

i. (3 pts) d->data int

j. (3 pts) a int

k. (3 pts) argv[1][2] char

6. Assume the `main` function in the program `runIt.cpp` starts with:

```
int main(int argc, char *argv[]) {
...
```

Further, suppose this program is invoked with the following command line:

```
./runIt mango grape
```

a. (2 pts) What is the value of `argc` in this case? 3

b. (2 pts) What is the value of argv[2][1]? r

c. (2 pts) What is the value of argv[0][0]? .

d. (2 pts) What is the value of argv[1][4]? o

7. (4 pts) Write a single line of code that declares an integer variable with the name `num1` and sets it equal to the value of `argv[1]`, converted to an integer.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

8. (4 pts) Write a single line of code that declares a variable for a real number (one such as 3.45) with the name `num2` and sets it equal to the value of `argv[2]`, converted as necessary. The variable should have an appropriate data type.

You may *assume* that any necessary `#include` directives already have been done, so you don't need to worry about that. Just write the line of code.

End of Exam
total points=100