

CPSC 327 Midterm

1. (3 points) Suppose you want to create a directory called 'delme' from your current location in linux. You open a Terminal window and then type what at the command line?

mkdir mydir

2. (3 points) In that directory you want to clone the following git repository <https://github.com/kperkins411/delme.git>
What do you type at the terminal command line?

git clone https://github.com/kperkins411/delme.git

3. (5 points) What is one advantage (in terms of safety) of using a reference when passing a parameter to a function verses passing it as a pointer?

A reference can never be null (0). So you do not have to check it for null like you do a pointer.

4. (5 points, multiple choice) Where does `vector.end()` point? What happens if you dereference it?

The last entry in the vector, you get the last entry if you dereference it.

One past the last entry in the vector, undefined behavior if you dereference it.

The last entry in the vector, undefined behavior if you dereference it.

One past the last entry in the vector, dereference safely returns the last entry

5.(5 points, multiple choice) For the following code. What does the preprocessor do and what is the preprocessors output?

```
//myfunc.cpp
#include "myfunc.h"
std::string myfunc()
{
    return "hello world";
}
```

Inserts contents of myfunc.h into myfunc.cpp. The output is pure C++

Inserts only the called portions of myfunc.h into myfunc.cpp. The output is pure C++

Inserts std::myfunc() into myfunc.cpp, The output is pure C++

There is a preprocessor error, so nothing happens

6. (5 points, multiple choice) For the following code. What does the compiler do and what is the compiler output?

```
//myfunc.cpp
#include "myfunc.h"
std::string myfunc()
{
    return "hello world";
}
```

Nothing, there is a preprocessor error

Takes the pure C++ and compiles it to myfunc.o (an object file)

Generates myfunc.exe (an executable) from the pure C++

Prints the message "hello world" to the console

7. (6 points, multiple choice) What is the difference between #include "myfile.h" and #include <myfile>

No real difference, you can use either. The compiler will find it.

The one with <> is a predefined system file, the compiler looks wherever predefined system files are kept, the one enclosed in "" appears to be a user file, the compiler will look in the users directory.

The one in <> is used for standard library headers so you cant put myfile.h in it. "" is for user files.

If the header file has a .h extension you put it in "". If not it goes in <>.

8. (8 points, multiple answer) What should go in a header file?

Include guards

Minimal number of includes so that the header compiles cleanly

variables

.cpp files

{ } after each function definition

All function declarations from the corresponding cpp file that you want exposed to other files

9. (3 points) How do you resize arrays in C++?

You can't they are sized at compile time. You could if you are using dynamic memory though.

10. (5 points) Please define an enum called fruit that contains Oranges, Apples and Pears.

```
enum fruit{Oranges = 1,Apples,Pears}
```

11. (5 points) What are the advantages to defining a variable with the fruit enum type as opposed to an int?

That variable can only be 1 of the 3 values (Oranges, Apples, Pears) and no other. If you use an int you can not restrict the assignment to 1 of those 3, any int will do, for ex

```
int apples=1;
```

```
int oranges=2;
```

```
int pears=3;
```

```
int myFruit=5; //what fruit is this? Cannot happen with enum
```










12. (6 points) Create a struct called fruitinfo with the following fields
Fruit Type (can only be Oranges, Apples or Pears)
numbPieces (an int)

```
struct fruitInfo{  
    fruit type;  
    int numPieces;  
};
```

13. (18 points) Please indicate what each cout will print in the space provided in the comment to the right.

```
void dopointer(){  
    int x = 3;  
    int *z = &x;  
    cout << "1." << *z << endl;    // __ 3 ____  
    *z = 14;  
    cout << "2." << x << endl;    // __ 14 ____  
    x = 12;  
    cout << "3." << *z << endl;    // __ 12 ____  
    *z = 28;  
  
    int* a = new int(37);  
    z=a;  
    cout << "4." << *z << endl;    // __ 37 ____  
  
    if (a)  
        delete a;  
    a=0;  
  
    char* A="012345";  
    char *B=A;  
    cout << "5." << *(B +2) << endl;    // __ 2 ____  
  
    cout << "6." << (B +2)[1] << endl; // __ 3 ____  
}
```

14. (20 points) You have a project with the structure given below. Please fill in all TODO comments in the following files. Please ensure you include the minimum number of include files necessary for the application to compile and link.

- ▼  midterm1
 - ▶  Binaries
 - ▶  Includes
 - ▼  src
 - ▶  constants.h
 - ▶  midterm1.cpp
 - ▶  utilities.cpp
 - ▶  utilities.h
 - ▶  Debug

```
// constants.h
#ifndef CONSTANTS_H_
#define CONSTANTS_H_

//TODO fill in includes if any needed

namespace KP_Constants{
    const int BINGO =3;
}

#endif /* CONSTANTS_H_ */
```

```
//midterm1.cpp
#include <iostream>
using namespace std;

//TODO fill in includes if any needed

#include "utilities.h"

int main() {
    func1(6);
}
```

```
//utilities.cpp
//TODO fill in includes if any needed

#include "utilities.h"
#include "constants.h"
bool func1(int i){
    //TODO return true if i equals BINGO in constants.h
    //false otherwise

    return (KP_Constants::BINGO == i);
}
```

```
//utilities.h
#ifndef UTILITIES_H_
#define UTILITIES_H_

//TODO fill in includes if any needed

bool func1(int i);

#endif /* UTILITIES_H_ */
```

13. (3 points) Create an vector that holds fruitinfo structs.

```
Std::vector<fruitinfo> myVector;
```